University of Alberta

Aggregate Production Modeling
Using Neural Networks and Belief Networks

by

Brent Warren Larison ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science

in

Construction Engineering and Management

Department of Civil and Environmental Engineering

Edmonton, Alberta
Fall 1998

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-34388-X

Canada

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled "Aggregate Production Modeling Using Neural Networks and Belief Networks" submitted by Brent Warren Larison in partial fulfillment of the requirements for the degree of Master of Science in Construction Engineering and Management.

Dr. S. M. AbouRizk

Dr. S.T. Ariaratnam

Dr. J.J. Cheng

Dr. J.D. Whittaker

Approved on 8/28/98

ABSTRACT

This research involves investigation of aggregate production simulation using an aggregate production simulation program called CRUISER. The research included onsite aggregate testing, simulation model testing, and program implementation. Neural networks and belief networks were two forms of artificial intelligence used to enhance the developed model. Large amounts of representative gradation data for crushers were simulated. Neural networks were then used to pattern this data after a cone crusher for eventual integration into the CRUISER program to model a specific crusher type and setting. A belief network was developed to semi-automate the optimization of the final product gradation for the user. The model can be used after each simulation run, after the gradation results provided by CRUISER are known. Procedures for using the developed belief network along with CRUISER are outlined and could be used in an employee training program or for educational purposes.

I would like to dedicate this thesis to my wife Susan.
For her love as well as unwavering support and guidance that has allowed me to pursue
and succeed in achieving this educational goal.

ACKNOWLEDGEMENT

I would like to thank my supervisor, Dr. Simaan M. AbouRizk for leading me to this research and for his time and direction. Appreciation is extended to Dr. S.M. Ariaratnam, Dr. J.J. Cheng, and Dr. J.D. Whittaker for serving as members on my examining committee. Many others have contributed to this research and helped me to attain my goals. Dany Hajjar has been instrumental in this research for assisting me with computer code, CRUISER modifications, and data collection. Thanks to Bill Laisse of Lafarge Construction who was very helpful with data collection and field input for this research. I would also like to thank Dr. Brenda McCabe for providing background and insight with respect to belief networks. Dr. Ashraf El-Assaly for assistance in editing and clarifying portions of this research work. The above people have all made contributions to this research in thought and insight, for which I am very grateful.

I would also like to thank my parents in law for their continued interest and support. Thanks to my parents, for without their support and encouragement throughout my education this accomplishment would not have been possible. Finally, I would like to thank my Lord Jesus Christ for the strength, wisdom and blessings granted to me.

TABLE OF CONTENTS

1.0 INTRODUCTION

- 1.1 Overview pg 1
- 1.2 Statement of the Problem pg 2
- 1.3 Research Objectives pg 3
- 1.4 Methodology of the Solution pg 3
  - 1.4.1 Aggregate Production Simulation - CRUISER pg 4
  - 1.4.2 Neural Networks pg 4
  - 1.4.3 Belief Networks pg 5
- 1.5 Thesis Organization pg 5

2.0 LITERATURE REVIEW

- 2.1 Introduction pg 6
- 2.2 Discussion of Aggregate Production pg 6
  - 2.2.1 Crushing pg 6
  - 2.2.2 Screening pg 10
  - 2.2.3 Equipment Selection pg 15
- 2.3 Accomplished Work on Aggregate Production pg 17
- 2.4 Accomplished Work on Aggregate Production Simulation pg 18
  - 2.4.1 Crushing pg 18
  - 2.4.2 Screening pg 20
  - 2.4.3 Mathematical Modeling pg 23
  - 2.4.4 Simulation Model Comparison pg 29
    - 2.4.4.1 Screens pg 30
    - 2.4.4.2 Crushers pg 31
    - 2.4.4.3 Conveyors pg 32
    - 2.4.4.4 Final Product Gradation pg 32
    - 2.4.4.5 Accuracy pg 33
    - 2.4.4.6 Overview pg 34
- 2.5 Discussion of Neural Networks pg 35

2.6 Neural Network Applications in Construction pg 39

2.7 Discussion of Belief Networks pg 44

2.8 Belief Networks in Construction pg 56

3.0 CRUISER VALIDATION

3.1 Introduction pg 57

3.2 Data Collection pg 59

- 3.2.1 Procedure pg 59
- 3.2.2 Samples pg 60
- 3.2.3 Analysis pg 60

3.3 Plant Characteristics pg 63

3.4 Product Sampling and Results pg 64

- 3.4.1 ACO Product Type pg 64
  - 3.4.1.1 Results pg 66
- 3.4.2 ACR Product Type pg 67
  - 3.4.2.1 Results pg 69
- 3.4.3 20mm Road Crush Product Type pg 70
  - 3.4.3.1 Results pg 71
- 3.4.4 TPH Analysis pg 72
- 3.4.5 Summary pg 74

3.5 Separate Component Analysis pg 74

- 3.5.1 Coarse Crusher pg 76
- 3.5.2 Fine Crusher pg 77
- 3.5.3 Scalper Screen pg 78
- 3.5.4 Coarse Screen – Upper Deck pg 79
- 3.5.5 Coarse Screen – Lower Deck pg 81
- 3.5.6 Fine Screen – Upper Deck pg 82
- 3.5.7 Fine Screen – Lower Deck pg 83
- 3.5.8 Results pg 83

3.6 Conclusion pg 84

4.0 SIMULATION ANALYSIS OF CRUISER

4.1 Introduction pg 86
4.2 Crusher Sensitivity pg 86
4.3 Crusher Analysis pg 88
    4.3.1 Crusher Setting pg 88
    4.3.2 Crushing Factors pg 90
    4.3.3 Lower Bound Factor – 0.5 pg 91
    4.3.4 Upper Bound Factor – 1.5 pg 92
    4.3.5 CRUISER Program Improvements pg 93
4.4 Screening Efficiency pg 94
4.5 Screening Analysis pg 98
    4.5.1 CRUISER Program Improvements pg 101
4.6 Conclusion pg 103

5.0 DEVELOPING A CRUSHING NEURAL NETWORK FOR CRUISER

5.1 Introduction pg 105
5.2 Crushing Neural Network – Prototype pg 105
    5.2.1 Sample Development pg 105
    5.2.2 Neural Network Architecture pg 108
    5.2.3 Optimum Training Parameters pg 108
    5.2.4 Neural Network Weights pg 109
    5.2.5 Neural Network Accuracy pg 109
5.3 Crushing Neural Network – Full Scale pg 110
    5.3.1 Sample Development pg 110
    5.3.2 Neural Network Architecture pg 111
    5.3.3 Preliminary Results pg 112
5.4 Crushing Neural Network – Trials pg 112
    5.4.1 Decreasing the Number of Gradients pg 112
    5.4.2 Emphasizing the Crusher Setting pg 113
    5.4.3 Cumulative Percent Retained pg 113
    5.4.4 Percent Retained on Each Sieve Size pg 114

    - 5.4.5 Increasing the Number of Gradients pg 114
  - 5.5 Crushing Neural Network – Final Model pg 117
    - 5.5.1 Neural Network Architecture pg 117
    - 5.5.2 Optimum Training Parameters pg 117
    - 5.5.3 Neural Network Weights pg 118
    - 5.5.4 Model Implementation pg 118
  - 5.6 Data Creation Procedure pg 119
  - 5.7 Implementation of Neural Networks pg 122
  - 5.8 Conclusion pg 122

6.0 OPTIMIZING PARAMETER SELECTION USING BELIEF NETWORKS

  - 6.1 Introduction pg 124
  - 6.2 Network Development pg 125
  - 6.3 Expert Information pg 128
  - 6.4 Simulation Information pg 134
  - 6.5 Belief Network Structure and Probabilities pg 135
  - 6.6 Testing the Belief Network pg 147
  - 6.7 Using the Belief Network pg 150
  - 6.8 Conclusion pg 156

7.0 FINAL DISCUSSION

  - 7.1 Summarized Findings pg 158
    - 7.1.1 Simulation Analysis pg 158
    - 7.1.2 Neural Networks pg 159
    - 7.1.3 Belief Networks pg 160
  - 7.2 Contributions pg 160
  - 7.3 Final Comments pg 161

8.0 REFERENCES pg 165

LIST OF TABLES

Table 2-1 Average Absolute Error Per Sieve Size pg 33

Table 3-1 Input Production Analysis pg 61

Table 3-2 Output Production Analysis pg 62

Table 3-3 Screen #1 – Screen Information pg 63

Table 3-4 Screen #2 – Screen Information pg 64

Table 3-5 Screen #3 – Screen Information pg 64

Table 3-6 Actual and CRUISER Simulated Data pg 73

Table 3-7 Coarse Screen – Lower Deck Efficiency pg 81

Table 3-8 Fine Screen – Lower Deck Efficiency pg 82

Table 4-1 Percent Passing Error of Lower and Upper Bound Factors pg 90

Table 4-2 Range of the Upper Bound Factor pg 92

Table 4-3 Screen Input TPH pg 98

Table 4-4 Efficiency Sensitivity pg 99

Table 4-5 Efficiencies Calculated by CRUISER pg 100

Table 4-6 Gradation Error Using Actual Screen Efficiency pg 102

Table 4-7 Calculated Efficiency with Accurate TPH pg 103

Table 6-1 Causal Node Descriptions and Corresponding Corrective Actions pg 137

Table 6-2 Probabilities for Too Coarse > 5mm Causal Node – Layer #1 pg 138

Table 6-3 Probabilities for Too Coarse > 5mm Causal Node – Layer #2 pg 139

Table 6-4 Probabilities for Too Coarse < 5mm Causal Node – Layer #1 pg 139

Table 6-5 Probabilities for Too Fine > 5mm Causal Node – Layer #1 pg 139

Table 6-6 Probabilities for Too Fine > 5mm Causal Node – Layer # 2 pg 139

Table 6-7 Probabilities for Too Fine < 5mm Causal Node – Layer #1 pg 140

Table 6-8 Probabilities for Too Fine < 5mm Causal Node – Layer # 2 pg 140

Table 6-9 Probabilities for High % Fracture Causal Node pg 140

Table 6-10 Probabilities for Low % Fracture Causal Node pg 140

Table 6-11 Case #1: Too Coarse > 5mm pg 143

Table 6-12 Case #2: Too Coarse > 5mm and Too Coarse < 5mm pg 144

Table 6-13 Case #3: Too Coarse > 5mm and Too Coarse < 5mm and High % Fracture pg 145

Table 6-14 Testing the Belief Network pg 147

Table 6-15 Suggestions for Different CRUISER Scenarios pg 155

LIST OF FIGURES

Figure 2-1 Plant Layout in CompuCrush pg 29
Figure 2-2 A Node – The Processing Element pg 35
Figure 2-3 Neural Network Structure pg 36
Figure 2-4 Training Phase of Network Development pg 37
Figure 2-5 Recall Phase of Network Development pg 37
Figure 2-6 A Simple Belief Network pg 45
Figure 2-7 Network Structure Comparison pg 47
Figure 2-8 A Belief Network for the Family-out Problem pg 48
Figure 3-1 Site Model (as depicted using CRUISER) pg 63
Figure 3-2 Actual vs CRUISER Results for ACO Product Type pg 66
Figure 3-3 Actual vs CRUISER Results for ACR Product Type pg 69
Figure 3-4 Actual vs CRUISER Results for 20mm R.C. Product Type pg 71
Figure 3-5 Actual vs CRUISER Results for the Coarse Crusher pg 76
Figure 3-6 Actual vs CRUISER Results for the Fine Crusher pg 77
Figure 3-7 Actual vs CRUISER Results for the Scalper Screen pg 78
Figure 3-8 Actual vs CRUISER Results for the Coarse Screen – Upper Deck pg 80
Figure 4-1 Various Crushing Parameters on Three Typical Gradations pg 89
Figure 4-2 Allis-Chalmers Screen Efficiency Curve pg 95
Figure 4-3 Distribution of Undersize in Oversize Based on Loading Ratio pg 97
Figure 5-1 Prototype Neural Network Accuracy pg 110
Figure 5-2 Sample Gradation Gradient Lines pg 111
Figure 5-3 Neural Network Accuracy – Trials pg 114

Figure 5-4	Neural Network Accuracy – Comparison	pg 115
Figure 5-5	Neural Network Accuracy – Final Model	pg 117
Figure 6-1	Initial Plant Layout	pg 127
Figure 6-2	Output Gradation as Observed in CRUISER	pg 135
Figure 6-3	Aggregate Production Belief Network	pg 136
Figure 6-4	Initial Plant Layout	pg 148

1.0 Introduction

1.1 Overview

The construction industry is responsible for consuming large amounts of aggregates. The primary uses of aggregates are for concrete production and road construction. Aggregate exists in nature in various sizes and shapes. Aggregate processing is required for aggregate to meet construction specifications, which are determined by the final use of the aggregate product. The two main processes involved in aggregate production are size reduction through crushing and size separation through screening. Size reduction is achieved through the use of specialized crushing equipment which operate on the principles of nipping (jaw, gyratory, and cone crushers) or high impact (single and double impeller impactors). The screening operation classifies the aggregate by grading the individual particles according to their minimum cross-sectional length. Typical screens contain a number of decks with varying opening sizes. Simple static screens transport the material across its surface by inclining the surface enough that the material is moved by gravity. Other screens transport material by utilizing a combination of gravity and rotating motion, which is supplied by vibration. The use of conveyors for transportation of the raw, intermediate and final products is quite popular since the particle size is smaller than the conveyor belt width in most cases. Despite the high capital cost, the use of conveyor transport is usually better than truck transport, since it has lower operating costs. Typically, decisions regarding plant changes and estimates of production are made by an experienced individual in the field. Through the use of computer modelling, an efficient and cost effective means of aggregate production decision-making can be achieved by personnel with limited levels of experience. The computer model used and on which this research is based is called CRUISER (Hajjar and AbouRizk, 1997). The CRUISER name is a short form of CRUshIng Simulation EnviRonment.

1.2 Statement of the Problem

The use of simulation in describing a problem and producing meaningful results can be quite costly and time consuming, depending on the task. Simulation is still a relatively new area that has been expanding along with the growing use of computers and data collection methods. The construction industry is quite traditional in nature and the use of new ideas and methods often conflict with preferred traditional methods, however inefficient or cumbersome they may be. It is difficult to quantify the dollar savings arising from the use of simulation up front without doing any initial work on a given problem. A substantial investment of time, resources, and patience is required before a working simulation model accurately reflects the actual processes. Any simulation program developed for the construction industry must be user friendly and must accomplish the desired task as easily as possible. Moreover, it must be accurate in its assessment of the actual processes being simulated so that conclusions about the behaviour of the simulated system can be relied upon with confidence. Simulation allows for inexpensive assessments as experimentation with the real variables occurs using the computer model. There is a challenge in developing a simulation model which is not too general yet not too specific so that it is practical and applicable to real life situations. Simulation is used to produce results of a set of variables in a given system, but is not often enhanced to optimize the results through the use of an automated process. Typically, the user must adjust the variables thought to significantly affect the result and keep track of the changes to the simulation results. Although there are many challenges and obstacles for simulation in the construction industry to overcome, the power it can demonstrate when developed properly can be very impressive.

1.3 Research Objectives

The objective of this research is to investigate the use of artificial intelligence in aggregate production simulation. Both neural networks and belief networks will add flexibility and additional modeling techniques to improve the simulation accuracy and use of the CRUISER program. Neural networks will be used to model a dynamic and unique aspect of aggregate production - crushing. The CRUISER program will be validated with actual data and inaccuracy will be improved primarily through the use of neural networks. Belief networks will add an optimization aspect to the program along with diagnostic features to enhance the various applications for use of the CRUISER program.

1.4 Methodology of the Solution

The methodology used to achieve the research objectives will be separated into three parts: The first part will outline the effectiveness of the existing aggregate simulation model and lead to program improvements. The second part will involve the development of a neural network to improve the crushing simulation portion of the model. The third part will include the use of belief networks to optimize the output gradation and serve as a diagnostic tool.

1.4.1 Aggregate Production Simulation - CRUISER

To demonstrate the current effectiveness of the CRUISER program and make program modifications to improve accuracy, the following steps will be taken.

1. Actual plant data will be collected, analyzed, and compared to CRUISER options and final output product gradation.
2. Components of the plant (i.e., crushers and screens) will be analyzed separately with the collected data as much as possible.
3. Further analysis of the crushing and screening processes within CRUISER will be evaluated and modified to improve simulation results from the program.

1.4.2 Neural Networks

To enhance the crushing portion of the CRUISER program, a neural network will be developed. Several options will be explored to obtain an accurate and yet realistic model for actual crushing data. The following steps were considered:

1. A prototype model will be developed to evaluate feasibility of the project.
2. Several tests with respect to neural network parameters will be evaluated; the data will be optimized to best represent actual crushing production situations.
3. A final model to enhance the crushing process with the CRUISER program will be made.

1.4.3 Belief Networks

To optimize the process of obtaining an acceptable gradation within the CRUISER program as well as develop a diagnostic and educational tool, a belief network will be developed. This will be done using the following approach:

1. Aggregate plant production diagnostics will be obtained from an expert.
2. A belief network will be developed using expert information while maintaining its ability to be used in conjunction with the CRUISER program.
3. The belief network will be tested and guidelines for its use will be developed.

1.5 Thesis Organization

Chapter 2 presents the literature review for this research. It covers the topics of aggregate production, neural networks, and belief networks. Chapter 3 presents the research involved in validating the CRUISER program with actual aggregate production plant data and analyzing the results. Chapter 4 contains further analysis of the CRUISER program pertaining to crushing and screening analysis. Chapter 5 presents a number of neural networks developed for the crushing simulation process within the CRUISER program. Chapter 6 demonstrates the use of a belief network to optimize the process of changing aggregate plant parameters in order to arrive at an acceptable final product gradation. Chapter 7 provides a conclusion to this research and identifies recommendations for future work with respect to neural networks and belief networks.

2.0 Literature Review

2.1 Introduction

The purpose of this literature review is to provide background information as well as insight into the purpose of this research. Background information on aggregate production and the maturity of this topic in simulation programs will be covered first. Aggregate production will be broken down into two main processes: crushing and screening. Two aggregate production simulation programs will be evaluated and compared. Second, background information and discussion of neural networks will be presented along with past research which has used this form of artificial intelligence. Third, the theory and background of belief networks will be presented and discussed.

2.2 Discussion of Aggregate Production

Mechanical size reduction is mainly carried out by crushing and grinding machines based on the principle of nipping the rock (e.g. cone, gyratory and jaw crushers) or of direct impact (e.g. impact crushers and hammermills). The particle sizing of fine and coarse material is based on the use of screens. Transportation of material to crushers, screens, and the product pile is achieved primarily with the use of conveyors. Some material hauling by a loader or dump truck is done within a crushing plant, but is usually kept to a minimum because of high costs.

2.2.1 Crushing

Crushing is usually a dry process, performed in a number of stages, with small reduction ratios ranging from three to six in each stage. The reduction ratio of a crushing stage can be defined as the ratio of maximum particle size entering the crusher to maximum particle size leaving the crusher. The breakage of particles is achieved mainly by crushing, impact, and attrition. The three modes of particle fracture (compressive, tensile, and shear) can be discerned depending on the rock mechanics and the type of loading. When a particle encounters crushing (compression failure), the products fall into two distinct size ranges. One size range of coarse particles results from tensile failure. The other size range results from compressive failure near the points of loading or by shear at projections of the particle. With impact crushing (tensile failure), a particle experiences a higher stress than is necessary to achieve simple fracture, and therefore tends to break apart rapidly. The resulting products are often very similar in size and shape. Attrition (shear failure) is due to particle-particle interaction, which results in large amounts of fine material. This can occur when a crusher is fed too fast and is usually undesirable. Crushing in closed circuit operations produce more undesirable fine material than do open circuit operations. The crushing action comes from stresses applied to rock particles by moving parts of the machine. The object of crushing in aggregate production is size reduction to a specified size range, with a minimum production of finer material. Energy requirements in crushing are relatively large, as the rock is essentially broken by compressive forces exerted on the material by the machine. Cone, gyratory, and jaw crushers achieve size reduction mainly by compressing particles between relatively slow moving, inclined surfaces. The material being fed into the machine enters from above, where the crushing surfaces are furthest apart, and is crushed into smaller fragments as it descends down into the narrowest zone of crushing and is finally discharged by gravity. The crushing surface in a jaw crusher consists of two rectangular plates, whereas in cone or gyratory crushers it is in the form of truncated conical shells. Jaw crushers usually consist of one fixed crushing face and an inclined mobile face, which moves a small distance back and forth from the fixed face. The major variables in jaw crushing are the angle of the jaws, rate of jaw movement, displacement of the mobile plate, and the distance between the jaws at the discharge end, which controls the product size. Gyratory crushers consist of an upward pointing solid rotating cone set within an inverted cone of varying angles. The inner cone rotates eccentrically so that it approaches and recedes from a given point on the outer cone during each revolution. Again, material is fed into the machine from above and is alternately compressed and released during the rotation of the inner cone. The eccentricity of the inner cone is arranged so that the apex maintains a fixed position, while the maximum displacement occurs near the bottom or the discharge area. Gyratory crushers have several advantages over jaw crushers in that they make more economical use of power and can handle wet and slightly clayey materials more effectively. However, jaw crushers can handle occasional oversize material while gyratory crushers cannot. Although gyratory crushers cost approximately three times more than jaw crushers, they are often selected because of their higher output rates. Jaw crushers are usually selected in cases where flexibility and intermittent use of the equipment become factors in the selection process. Cone crushers are similar to gyratory crushers except that the outer crushing surface is in the form of an upward pointing apex, which results in the two crushing surfaces being nearly parallel to each other. This design allows the crushed material to spread out as it works its way downward while preventing close packing and blockages in the discharge area. There are a large variety of different crushing surface geometries; these depend on the particular desired aggregate requirements. The two crushing surfaces are held together by springs so that they can separate under a load to allow any tough oversize material to pass through without interrupting the crushing process. These characteristics, together with a high speed of rotation, make the cone crusher highly suitable for achieving high throughputs of intermediate particle size where a limited reduction in particle size is required. Typically, cone crushers are used sometime after jaw or gyratory crushers (i.e., for secondary crushing). Cone crushers do have one significant disadvantage; they tend to produce more flake-shaped particles than other crusher types.

Impacting machines are also widely used in aggregate production. Hammermill type crushers are characterized by a fast moving rotor with attached beaters or hammers which use a striking action to break the rock. Internal stresses may then cause the particle to shatter immediately or it may be sent, with broken fragments, against an impact plate. Further shattering may occur when the particle or particles hit the impact plate. Various sizes, as well as different discharge and rotor configurations, can be found. The maximum product size is controlled by the distance between the hammers and breaking plates. Sometimes this distance varies to provide room for successive breakage of particles. Advantages of impacting machines include lower energy requirements, an ability to handle high proportions of clay or shale, a tendency to produce more cube-like particles, the capacity to achieve a large reduction ratio, and small space requirements. The one major disadvantage of impact crushers is that tough abrasive fragments cause intense wear on the moving parts, which results in a high cost. Therefore, they are usually used for crushing limestone, which is not very abrasive. Sometimes they are used in situations where their tendency to produce minimum amounts of elongated particles is of some value in meeting the specifications with difficult materials.

The presence of significant amounts of fine material in the feed stream to a crusher decreases the tonnage of coarser material that can be processed. A steady continuous supply stream is important for the optimum performance of a crusher. This is usually accommodated with the use of a surge bin at the raw input stage in an aggregate processing plant. Sometimes intermediate surge bins are required when the large raw product particle sizes are too large for a single surge bin to handle.

2.2.2 Screening

Most aggregate sizing operations are carried out with the use of a screening operation which grades particles according to the minimum cross section presented to a wire mesh or some other gradation device. The rate of material passing through a gradation media is directly proportional to the hole size which is presented to the material. The types of screens that process material can vary in the method they use to transport the material across the gradation media. Simple static screens transport the material by inclining the surface sufficiently for the material to be moved by gravity. Other inclined screens utilize gravity along with vibration to transport the material. Another type of screen is an inclined cylindrical screen, which transports the material by gravity and rotary motion. Other screens, know as horizontal screens, transport material through the use of vibration only. There are two reasons for screens to become blocked and therefore less efficient in screening material. The first is when single particles become wedged in the screen openings. This occurs most often when a high proportion of material with diameters between 70 and 110% of the screen size is present in the feed. This situation can be minimized through the use of thinner, rubber or flexible wire screens. Adjusting the vibration rate of a screen can also help with this problem. Another situation is where fine particles adhere to each other and block the screen openings. This is usually due to excess moisture or clay particles in the material and is typically resolved by prewashing the material or screening under a water spray. Screen openings are normally square, circular, or slotted in shape. Mostly square screen openings are used, but circular and slotted openings are used for special applications. Circular openings are good for screening angular or flaky particles, whereas slotted openings are good for screening finer particles. However, in most circumstances, the selection of the shape is secondary to the selection of the right opening size for a screen.

Recently other material surfaces have been sought for screens as a substitute for steel. These are usually rubber or some other polymer material and are available with or without steel or fabric reinforcement. Although rubber screens can cost up to three times more than steel ones, they can last up to ten times longer. Another advantage of using these substitutes is that they have a lower level of operating noise, which is becoming increasingly important due to current health and safety regulations. Polymers are typically weaker than steel and require more supporting material, thus increasing the screen wire diameter and reducing the amount of effective screening area for aggregate to pass through. As a result, polymer screens are used where abrasive material warrants their use to offset maintenance costs.

The functions of screens are: 1) to protect crushers by controlling the material size of the feed and 2) to grade the crushed material into specific size ranges. The design of a screening layout depends upon the optimization of several factors. These factors are economic and mechanical in terms of size distribution to be processed and the markets in which the aggregate will be sold to. There are a number of screen types that are commonly used in aggregate production. The most widely used type is the inclined vibrating screen. Vibration is produced by a single eccentric driving shaft and the angle of the screen can vary depending on the direction of material flow across the screen. Angles of up to 20 degrees are used if the direction of motion is the same as the transport direction. Occasionally angles of 20 to 30 degrees are used if the motion is opposite to the material flow and a more vigorous screening action is required. The inclination of the screen is very important because too much slope results in material flow across the screen occurring too quickly for the undersize particles to pass through the openings, while too little slope results in a reduced screening capacity. The use of multiple decks on one screen does not provide the optimum screening process, but separate screening machines are too costly and therefore multiple deck screens are often used. Usually, screen decks are divided into splits that allow for the use of different screen sizes on the same deck.

Sometimes finer screen sizes are used at the feed end of the screen and coarser sizes at the discharge end to regulate the gradation more precisely.

A second type of screen is the horizontal vibrating screen. This screen type is often sloped up to 5 degrees and makes use of linear motion, which stratifies as well as transports aggregate along the screen deck. Horizontal screens are more costly than inclined screens but have more capacity and are lower in stature. Either advantage can come into play when contemplating the use of a horizontal screen. A third type of screen is the trommel screen. It consists of a slightly inclined cylindrical screen that rotates as material is passed through it. Material enters the trommel at the higher end and is discharged as undersize underneath the screen or as oversize at the end of the screen. Advantages of trommels are that they are strong and cheap, and that they utilize no vibration for screening. However, they have a poor capacity for their screen surface area and changing the screens can become difficult. They are still used intermittently and for washing applications. A fourth type of screen is the grizzly screen, which consists of parallel bars of various cross sections held together by widely spaced bars. They are very tough and are mainly used for initial screening of the untreated raw product. They may be inclined to allow the oversize material to slide off the screen, or kept horizontal to act as a device to break large rock fragments. Vibrating grizzlies are often used to feed coarse material into a primary crusher and separate undersize particles that do not require processing or are unsuitable for the final product. This material can then be treated as waste, sold as-is, or further processed as required. Screens that perform the initial screening of the raw product are often referred to as scalper screens because they remove the waste or scalpings from the final product.

Another type of material separation involves the use of classifiers and is typically used to remove finer material. These can be of either a dry or wet type but usually have some size of tank for receiving a slurry feed. Fine particles are carried upwards with the flow of water and are discharged over a weir, while coarser material settles to the bottom where it is discharged from the tank in a variety of ways. It can be done through a valve in the base of the tank, by elevated buckets, or by a spiral or rake moving along the bottom of the tank on an incline. The size at which separation occurs is a function of the height of the weir and the flow rate of the water.

Another machine sometimes used in the classification process is the cyclone. This machine is mainly used in sand treatment plants for the rejection of material finer than 75 microns and for dewatering. It is made up of a hollow inverted cone with axial discharge points and an inlet near the top. Slurry enters into the cyclone through the vertical axis and a vortex about the vertical axis sends coarse particles to the wall of the cone where they encounter a downward flowing zone that directs the material to the discharge point. The finer particles move in an upward flow towards a tube, which then extracts them from the cyclone. Cyclones are relatively cheap and small, and have few mechanical parts. Theoretically, they are complex in that the size at which they separate is governed by factors like feed rate, feed pressure, feed inlet diameter, and outlet diameter.

Therefore individual cyclones are optimized for a particular duty within an aggregate production plant and are more commonly found in industrial aggregate processing plants.

2.2.3 Equipment Selection

The selection of equipment for the design of an aggregate plant is not an easy task. There are some general principles to follow with respect to the procedures most suitable for particular materials. Impact machines are mostly used for processing limestone or other plastic materials. Sometimes it is necessary to process hard material using impact machines just to obtain a more cube-like particle shape. For harder materials, a jaw crusher or gyratory crusher is typically used to carry out the primary crushing process. Gyratory crushers are mostly used for large continuous operations. For secondary crushing, cone crushers are used because they can maintain high throughput due to their requirement of a uniform top size of feed. The only problem with cone crushers is their ability to produce some flaky particles, which can occasionally cause problems. The greatest control of particle size and shape is obtained by restricting the achieved size reduction in every crushing pass of the material. The restriction of having small size reductions for each crushing pass is the increased number of crushing stages needed for a given plant. The benefits of each crushing component must be justified against their high initial costs as well as operating and maintenance costs. The capital cost per tonne of product made at the same closed side setting does not decrease significantly with crusher size.

When selecting a crusher, the most important factors are the required capacity and expected feed size. Screens are used to remove finer material from the product as well as control the material flow to crushers. The finer material is usually the weaker portion of the final product and therefore it is desired to minimize its presence in the final product while yet retaining some for volume purposes. Screening is also used to reduce the proportions of flaky or elongated particles in order for the product to meet specifications. When processing sands, low density materials with relatively low particle velocity in water can be removed using cyclones or classifiers. Since particle velocity depends on size as well as shape, cyclones or classifiers are only effective for sands where quartz grains and undesirable flaky material are of the same size. Therefore, for optimum results, it is preferable to screen the sand into relatively similar size fractions before classification. Wet processing is necessary when the moisture content of the material is too high or the level of clay content would cause plugging in a dry screening process. Wet operations involve the additional expense of providing a suitable water supply, dewatering equipment, particle settlement, and water circulation. Stationary plant layouts vary from operation to operation because the nature of the natural source is rarely the same from one location to another. Sometimes the need for the plant to be mobile or the use of existing equipment governs the equipment layout of the plant. These plants usually sacrifice optimizing the aggregate production process to some degree, but the cost effectiveness of their operations still warrants their existence. Plant design usually involves laboratory testing along with several trials during the initial plant setup stages to accurately determine the value of an aggregate crushing operation.

2.3 Accomplished Work on Aggregate Production

There is ongoing research in Europe pertaining to modifying the crushing process to obtain higher quality aggregate, thus counteracting the problems of a high usage of studded tires and winter freeze-thaw cycles (Heikkila, 1992). The trade off of higher production costs and profit against better quality aggregate could not be determined, since the research could not affect the aggregate production schedules. The research pursued three objectives: 1) to determine the existing quality obtained using current crushing processes, 2) to test ways of improving the quality, and 3) to suggest ways to improve the quality of aggregate production with only minor changes in the equipment being used.

Major findings in this research were:

1. Blasting of rock should be done more carefully to reduce the amount of poor aggregate generated by this necessary process, therefore improving the quality of the final product. However, the improvement in quality resulted mostly from changing crushing processes instead of more careful blasting of rock.
2. Poor strength and badly shaped particles from blasting and primary crushing should be removed from the final product. When top-quality is desired, it is necessary to remove the product from the intermediate crusher as well. Doing this improved the quality dramatically and proved to be a feasible change for most aggregate plants.
3. Compatibility and not availability should govern the selection of crushers.
4. Regular crusher maintenance is essential for producing good quality aggregate.

5. A closed crushing circuit for the last crushing stage dramatically improves the quality of aggregate.
6. The best shaped properties in the crushed product are those in the particle size fraction closest to the setting of the crusher.
7. The highest quality of aggregate was obtained from the product of the final crushing stage instead of the final product of the entire plant.

2.4 Accomplished Work on Aggregate Production Simulation

2.4.1 Crushing

Some simulation work has been done on the area of mineral crushing and processing plants. Some of the same processes and equipment are used for aggregate production processes as well and will form a portion of the discussion in this section. Early research into the understanding of the comminution process was concerned with the relationship between the energy consumed and the size reduction attained by that energy. This consideration of energy input as a function of the grinding system was very attractive at the time, but was much more complicated than originally realized. Not all of the expended energy is spent in the breakage of particles only; forms of friction and sound also require energy. Comminution is basically considered to be a result of a mechanical operation that consumes energy and indirectly achieves a reduction in particle size. As outlined by Lynch (1977), it is generally considered that energy size-reduction relationships are not suitable to define the process of size reduction due to the difficulty in measuring energy losses in the form of friction and sound.

One approach developed by two groups of researchers (Gurun, 1973) and (Canalog and Geiger, 1973) used stored performance data supplemented by simple empirical relationships to construct models of unit processes and complete circuits. These models do not take into account the changes in the performance of the machines under different loading and operating conditions. Nor do they allow for an optimum design or consideration of future expansion. Another approach involving accurate mathematical models of unit processes developed from plant data had limitations as well. According to Lynch (1977) these models are only useful when expanding an existing plant or designing a new plant for processing material similar to that on which the model is based. Parameters in these mathematical models can be estimated from laboratory or pilot-scale data, but presently it is not possible for this to be achieved with sufficient accuracy.

There have been only a few attempts to model the crushing process with mathematical models. One basic underlying mechanistic model, proposed by Epstein (1948), showed that the distribution function after a number of steps (N) in a repetitive breakage process can be described by a probability function and a distribution function. This concept has led to the development of matrix and kinetic models. The matrix model considers comminution to be a succession of breakage events, whereas the kinetic model considers it to be a continuous process. In the matrix model, one part of the underlying theory is that particles in all size ranges have some probability of falling into any smaller size interval. This model requires the development of these probabilities from actual data. The probabilities may change as the size of the particles change, making it increasingly difficult for the model to be useful. Another part of the theory is the development of a breakage function, which is used to describe the product of a single breakage event. This function has been very difficult to determine experimentally because there is no non-destructive testing technique which will give information about the inherent breakage properties of minerals. In the kinetic model, comminution is considered to be a rate process expressed in terms of continuous functions and discrete distributions. As outlined by Lynch (1977), the major difficulty in the application of the continuous distribution model to practical problems is that of obtaining a satisfactory definition of the continuous function for the distribution of particle sizes.

2.4.2 Screening

There have been a couple of attempts at developing mathematical models for screen operations for use in simulation studies. Gurun (1973) developed a representation of screen behaviour by a column vector in which successive elements described the probability of each fraction appearing in the oversize. For any given case, the column values were chosen from a set of equipment and performance data. This model becomes useful when predictions about screen behaviour are available and correct, but are limited in other instances. For example, the model would have difficulty predicting screen behaviour when a screen is operating under a fully loaded or overloaded condition. Other work was done by Whitby (1958) to predict screen behaviour, but this involved batch screens only. Batch screens are where particles are placed on the screen and vibrated for a period of time before being removed from the screen and replaced with another batch. While there is a close relationship between batch and continuous screening, the adaptation of Whitby's model for simulation purposes would require considerable amount of special data collection from continuous screens; this has not yet been done.

A more recent application of simulation in a mineral processing environment was done by Du Plessis (1994). In this study, a system simulation was done where different process units are linked together and interact with each other. Each process unit performs a function separate from the entire system, but on a much smaller scale. Within the model, each process unit is a subroutine of code. The primary simulation model used in this research is called Siman 4. This simulation package performs discrete simulation that utilizes entities and attributes which describe those entities. Historical data from an actual plant was collected to determine the size distributions for the feed and product streams for the screen and crushers at different settings. These samples were collected by stopping selected conveyor belts and obtaining a 2 meter long section from the belt on which a sieve analysis was done. The feed rate of the raw product was variable, so it was modeled with an exponential distribution describing the probable arrival times of entities into the simulation system. Experiments were also carried out on the plant to determine the correlation between energy consumption and tonnage handled by the screens and roll crushers. Flow rates were determined by taking weightometer readings during plant operations.

The simulation model for the screens is based on the Karra model (Karra, 1979). Karra's model is a predictive one; it describes screen behaviour using capacity factors which depend on the tonnage, and size distribution of the material fed onto the screen, as well as the characteristics of the screen itself. This model is entirely empirical. The simulation model for the crushers is based on the classification and breakage model developed by Whiten at the Julius Kruttschnitt Mineral Research Center. This model was designed to simulate the crushing process for mineral processing applications. It does not account for varying feed rates, which affect the final product size distribution. This limits the applicability of this model to certain operating conditions. The researcher verified the accuracy of the model and used it to develop alternative plant operating strategies. The plant actually implemented some of the changes suggested by the researcher and verified the results of the anticipated changes. This research demonstrated the use of a general purpose simulation model with animation for evaluating the operating performance of a metallurgical plant.

Another application of simulation for mineral processing was done by Duursma (1990). In this research, an iron ore benefaction plant was evaluated using a simulation package called Microsim. Some crushing and screening models were studied, developed and enhanced. The Whiten crushing model was modified for hematite and a new model was developed for gyradisc crushing of iron. A Nordberg crushing model was evaluated with actual plant results and was found to be less accurate than the Whiten model. The Karra screen model was enhanced to include other deck types and non-square apertures to properly model the existing plant. The simulated crushing and screening processes were specifically tailored to processing of iron ore. This was done using previous research on mineral crushing along with screening data gathered in many conditions over several operational years of the plant.

This model relies on the user to ensure that screen efficiencies used for a particular simulation apply for the feed rates calculated after a simulation run. The screening models developed rely on either empirical data, or on both empirical and experimental data. The simulation model was accurate in predicting output flow rates within 1% but was less accurate in predicting the gradation for some portions of the plant. The researcher attributes this to the inaccuracy of the gyratory crushing model or the variability in material flow rates occurring in the actual plant. The main problem with this model is its inability to accurately simulate variable feed rates; this problem is common at ore processing plants. A second limitation in the use of this model is that it can only be used by an experienced Microsim operator. The researcher concluded that Microsim should not be used to simulate crushing circuits that may contain unsteady flow conditions. It is noted as well that the accuracy of the model also depends on the quality of data used as input for the model. Overall, this research resulted in an accurate steady-state crushing model applicable for an iron ore benefaction plant. The screening models were enhanced to model a particular plant, but are easily transferable to other mineral screening operations. The researcher demonstrated that the developed simulation model could be used as an aid in solving ore dressing production and design problems.

2.4.3 Mathematical Modeling

A mathematical model of aggregate production for simulation purposes was developed in the early 1970's (Hancher and Havers, 1972). The goal of this research was to provide an analytical model to evaluate the production characteristics of an aggregate production plant. It was intended for use by industry personnel both for the design of new aggregate processing plants and for the analysis of existing plants. The model has been developed with a number of subroutines, each analyzing a specific aspect of the plant. The output from the model predicts the flow rate and final product gradation for each material stream in the plant, whether it is for a conveyor or product pile. The screening process is modeled through the use of empirical data and equipment manufacturing specifications. The capacity of a screen deck is determined using a formula which is based on equipment manufacturing specifications. The efficiency of a screen deck is modeled by empirical data created in the 1960's by the Allis-Chalmers Manufacturing Company. The actual capacity of the screen deck is determined by adjusting the calculated theoretical capacity by using the empirical efficiency curve. This actual capacity affects rates of material flow from the screen deck as well as the oversize and undersize product gradations. The crushing process is modeled through the use of equipment manufacturing data, which describes output gradation depending on the crusher type, model, and selected crusher setting.

This research was the first valid aggregate production simulation model found for general application rather than exclusively for specific industrial plant designs. This model contributes to more effective plant utilization and plant design. The model can also be used to evaluate the maximum production capabilities of existing plants and assist in identifying the plant components that limit the production capabilities of a given plant. The model should also assist the plant operator in arriving at values for plant variables, such as crusher settings and screen mesh sizes, which are required to obtain multiple product output from a single plant. However, the predictions of the model should provide reasonable initial estimates for the plant variables, and thus reduce the field adjustment process. The researchers verified the model by comparing simulation results with those obtained from a number of limestone processing plants. Sampling performed at each plant was only sufficient to provide an approximate estimate for the plant output. A rigorous study of the fluctuations in the output characteristics for each plant would involve the collection and analysis of a large number of samples, which was not feasible at the time the research was conducted. The available data pertaining to the production capabilities and related characteristics of aggregate plant equipment were essentially confined to limestone processing. Therefore, the capability of the model to provide acceptable aggregate processing predictions for aggregates whose properties are significantly different from those of limestone are unknown.

The researchers obtained satisfactory results from the computer model for the plants discussed. This might be attributed to a good initial estimate of the raw feed contents, and to the fact that the plant is a limestone processing facility, the type for which the predictive data is most applicable. The predictions for the larger particle sizes varied substantially more from the sampled results than did the predictions for the smaller sizes. The likely reason for this is due to the sampling process. It was difficult to obtain a uniform sample from the screen, both because of the sizes of the larger particles and because of the fast flow of material across the screen. It is also probable that a greater number of samples should have been taken for accuracy analysis. For two out of the three plants, reasonable agreement was found to exist between the predicted results and the observed results for most of the processing streams. Excluding the raw feed streams, the predictions for the 14 of the streams sampled were consistently within 20% of the observed results. Appreciable variation could logically be attributed to sampling difficulties. The apparent failure at the third plant was attributed to two causes: First, some of the particle sizes in the raw feed stream were so large that it was not feasible to sample or measure them, so they were estimated by the plant superintendent. Second, there was a reversible impactor crusher, for which the equation used was not considered satisfactory.

Overall, the model has been developed as a system of subroutines, each one of which performs a specific task in the analysis of a plant. It provides a simple format for setting up a plant analysis; it allows the model to be programmed for small computers; and it facilitates updating or extending the basic production model. It should be noted that available data pertaining to the production capabilities and related characteristics of aggregate plant equipment, which were used to formulate the input-output relationships for the simulation model, were largely derived from the industrial experience in processing limestone materials. Therefore, the capability of the model to provide acceptable aggregate-processing predictions for aggregates whose properties are significantly different from those of limestone are unknown.

One improvement that could be made to assist in industry's general adoption of this model is to have an interface which would make the model more user friendly. Understanding when this model was developed leads us to believe that an improvement in this area would have been investigated if it were possible to do so at the time. Another significant improvement would be to enhance the modeling of crushers to increase the accuracy of modeling this process. The data provided by equipment manufacturers is somewhat crude and inaccurate.

A third improvement to this model pertains to circulating loads in a closed-circuit aggregate plant operation and would assist the user in proper equipment selection and plant design. This model does not sufficiently address the issue of circulating loads in a closed-circuit operation. When a closed-circuit plant is first put in operation, the circulating load will build up, quickly at first, then more gradually as the material begins recycling through the plant. A maximum or "steady-state" value of the circulating load will ultimately be reached if the plant equipment in the closed-circuit is of adequate capacity, and if the raw feed rate is reasonably uniform. The user must estimate the expected size of the circulating load with reasonable accuracy if the plant is to be designed efficiently. The circulating load in a closed-circuit aggregate plant operation is a significant factor in equipment selection and plant design. A poor estimate of this load can lead to over-design and unnecessarily high investment costs, or to under-design and, again, unnecessarily high operating costs.

A fourth improvement would involve enhancing the screen capacity calculations, thus allowing the model to accurately predict the actual capacity of screens. The screen capacity calculations are somewhat sensitive to variable changes. This is because of the method used to predict screen efficiency; there are only minor changes in the size separation predictions, unless the screen is considerably overloaded. The predicted screen efficiency does not drop below 65 % until the actual load exceeds 180% of the rated capacity. Also, for screen loading ranging between 50% and 135% of the rated load, the predicted screen efficiency only varies from 80% to 95%.

These are not the only improvements that could be made to the model, but they are the major ones required for improving the accuracy and acceptance of the model. Even without these aforementioned suggestions the model is a significant improvement over those preceding it. This model was the basis on which the CRUISER program was developed (AbouRizk and Hajjar, 1997). A user manual for the CRUISER program can be found in Appendix A. The user interface of the CRUISER program utilizes a visual object oriented environment, which makes the model much more user friendly. The modeling of the crushing process is enhanced by this research through the use of a form of artificial intelligence called “neural networks”. The modeling of circulating loads has been enhanced within the CRUISER program by the process of looping through the necessary subroutines that represent the given plant layout. The simulation of the closed-circuit is completed when a relatively constant output load from the closed-circuit portion of the model is achieved. This research has improved the screen capacity evaluation by doing two things: First, factors within the model responsible for the screen capacity calculations were adjusted to reflect actual collected data more accurately. Second, the user is now able to choose between the empirical approach for the screen capacity evaluation and inputting their own educated prediction of the screen efficiency for a specific deck on a particular screen within the model.

2.4.4 Simulation Model Comparison

A simulation model, similar to CRUISER, was developed by Cedarapids Inc. (Cedarapids, 1994). It is called CompuCrush©. This simulation model utilizes a visual object oriented environment in which a plant layout can be modeled and gradation output can be determined when samples of input are given. The methodology of how the simulation model works is expected to be somewhat similar to that of CRUISER. However, a test to see how this software compares to CRUISER will be done to evaluate its comparative performance. A view of the interface and plant modeling is shown in Figure 2-1. Additional screens of CompuCrush along with comparable screens from CRUISER can be found in Appendix B.

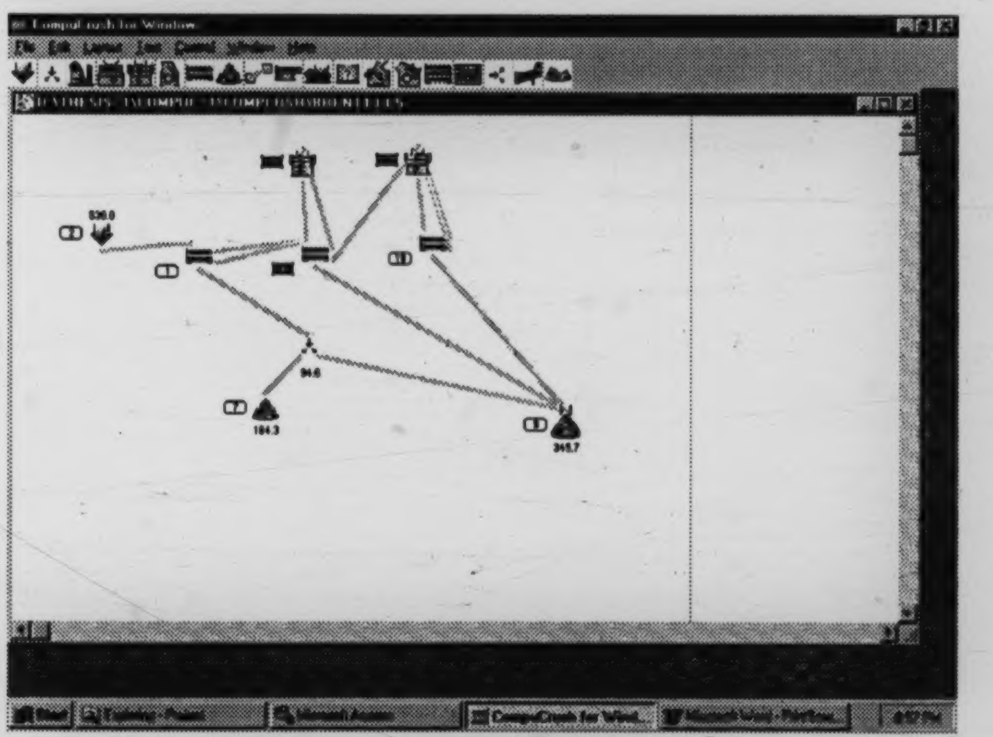


Figure 2-1 Plant Layout in CompuCrush

There are a few main differences between the layout options found in CompuCrush and those found in CRUISER: CompuCrush has additional icons representing a VSI crusher, HRSI crusher, grizzly, hammermill, washer/classifier, and a generic cone. Of these additional icons, the grizzly and generic cone options are not really necessary but are convenient for visually modeling an actual plant. The grizzly is a simple slotted screen used to screen fine material before a jaw crusher. The generic cone allows a user to model his/her own brand of crusher, which may not be of a Cedarapids brand. This is convenient, but the input for this cone is just a gradation for a specific closed side setting of the crusher. This is characteristic of the pocket reference handbook, but does not model different material types, nor other variables affecting the gradation. CompuCrush also has a text box for the user to insert comments on the screen, which will be displayed on the layout of the plant.

2.4.4.1 Screens

In CompuCrush there are only three deck screens allowed, whereas CRUISER has up to four decks for the user. CompuCrush has only two splits for each deck for the user to configure options for. CRUISER has up to four splits for each deck if the user desires to use them. Although CRUISER has more options as far as the number of decks or screens available, the more limited options pertaining to screens within CompuCrush will be sufficient in most cases. CompuCrush has only three different wire types, whereas CRUISER has a total of five different types. This difference between the two is not very significant; the three specified in CompuCrush are typically used, and the type of screen used on a screen has a lesser effect on the gradation than other screening variables (i.e., screen size).

One major difference between screening processes is how each model utilizes screening efficiency. Within CompuCrush, the default screen efficiency for each deck is 90%, and the user has the option of manually changing this if so desired. However, the screen efficiency default within CRUISER is a calculation based on screen loading for each deck during the simulation process. CRUISER also provides the option of letting the user define a desired efficiency for each deck, as does CompuCrush. CompuCrush does provide the option for the user to view the factors for the screen that the model uses as based on the options the user selected while configuring the screen. This is useful in that the user can further evaluate the screening calculations and determine which factors might be affecting the output gradation of the screen. Instead, the response of a screen to various factors could be evaluated through a trial and error process, but this leaves the user somewhat uninformed unless he/she understands the information in the Pocket Reference Book provided by equipment manufacturers. CompuCrush supports a washer/classifier type of screen, whereas CRUISER does not. Although this screen is limited in its available options, it is based on the Pocket Reference Book and is typical of the more common applications for this screen.

2.4.4.2 Crushers

With respect to cone crushers, CompuCrush has the option of selecting either a standard, fine, or sand type of conehead for simulating the crushing process. CRUISER does not have this option of conehead type. Impeller, jaw, and roll crushers are modeled the same way in each model. CompuCrush does model three other types of crushers: Vertical Secondary Impactor (VSI), Horizontal Rotor Secondary Impactor (HRSI) and a hammermill crusher. A HRSI crusher is typically used to process limestone material. A hammermill crusher is usually a secondary crusher used for processing high quality limestone. These three crusher types are not typically used in aggregate production in Canada. They are, however, commonly used in mineral aggregate processing and limestone production in the United States.

2.4.4.3 Conveyors

Conveyors are used in each model to connect equipment components and direct the material through the plant. CRUISER uses the conveyors to visually demonstrate the direction of material flow through the plant, whereas CompuCrush does not. Another difference between the two models is that CompuCrush determines the tonnes per hour (TPH) capacity of the conveyor based on user information. This information consists of belt width, roller type, and the speed of the conveyor in feet per minute (FPM). For CompuCrush to calculate the estimated horsepower required for a chosen conveyor the user must supply the model with the overall length of the conveyor and its elevation angle. CRUISER does not calculate the capacity for conveyors, nor the required horsepower to operate them. Both models allow for the user to view the gradation of the product on every conveyor in the model once the simulation has been run.

2.4.4.4 Final Product Gradation

CompuCrush has two options for viewing the gradation output: One involves viewing a line graph with the values as % passing or % retained. The other option is to view the gradation by way of a histogram in either % passing or % retained. CRUISER gives the option to view the gradation in the same manner as CompuCrush, with the exception that there is no option to view the results in the form of a histogram.

2.4.4.5 Accuracy

The final product gradations were evaluated for a comparison between the two aggregate production models. A total of seven different samples that were collected through this research was compared between the two models. Graphs of the actual output, CompuCrush, and CRUISER gradations in percent passing format can be found in Appendix B. The actual gradation was obtained from plant testing, and the parameters for both models were put to the same settings. The error for both models as compared to the actual and then to each other can be found in the table below. One sample type is called ACO, which stands for Asphalt Concrete Overlay, and is a product used for pavements put over existing pavement. Another sample type is called ACR, which stands for Asphalt Concrete Residential, and is a product used for pavements in residential areas. The last product type sampled is called 20 mm Road Crush, and is typically used for road base underneath asphalt pavement.

Table 2-1 Average Absolute Error Per Sieve Size

Sample Type and Number of Samples	CompuCrush	CRUISER	Error Ratio of CompuCrush to CRUISER
ACO (3 samples)	9.9	6.2	1.7
ACR (3 samples)	11.1	5.0	2.3
20mm Road Crush (1 sample)	9.6	7.6	1.3

It is clear that CRUISER is more accurate than CompuCrush in predicting the final output of an aggregate plant. It is shown in Appendix B that CompuCrush generally predicts a more coarse product than the actual gradation, while CRUISER generally predicts more fine than the actual gradation. The only exception to this is the 20mm Road Crush product type where CRUISER is also coarse. It can be said that the accuracy of CompuCrush comes closest to CRUISER when a coarser final product aggregate is being simulated. This could be due to the fact that CompuCrush is designed more so for industrial aggregate processing than CRUISER. Industrial aggregate processing typically handles larger aggregate sizes and utilizes different types of equipment. It is the conclusion of the researcher that CRUISER is more accurate because of the difference in the screen modeling process. The modeling of crushers is identical for both models at the present time, and is therefore not a factor in how the final product between each models deviates from each other.

2.4.4.6 Overview

In general, the two models are quite similar in that they are both object oriented and work out of a Windows© based environment. They also handle raw data input and display product output in very similar ways. The visual display of the objects with the CRUISER model is better than that of CompuCrush. How the two models describe a given plant with various components is quite similar. How the equipment within each model is configured is done in much the same manner. CompuCrush does model more screen types than does CRUISER although the screen modeling options are similar. The CRUISER model does allow for user input for screen efficiency calculations whereas CompuCrush does not. CompuCrush contains more crusher types than does CRUISER. CompuCrush allows for more descriptive information to be added to the model and has better reports for results than CRUISER. With the samples obtained in this research CRUISER is more accurate in predicting the final product gradation than CompuCrush. In overall user friendliness both models are quite good.

2.5 Discussion of Neural Networks

Neural networks are a branch of artificial intelligence. Neural computing is an attempt to model the processing power and functionality of the human brain using a computer. The functions of the human brain modeled by neural networks are the problem solving and memory functions. The basic element in a neural network is a processing element, modeled after the basic unit of the human brain, the neuron. Each neuron is a simple processing unit; when connected to each other they form powerful processors. Every neuron or node takes several inputs simultaneously and adds them, resulting in a response dependent on the level of inputs received. The reaction of the node depends on whether the sum of the inputs is high or low. The sum of the input values is then modified for output by means of a transfer function. The transfer function can be a threshold function, which only passes on information if the combined input reaches a particular value. It can also be a continuous function, which allows emphasis to be placed on certain input values. The transfer function might add weight to high value patterns and ignore low ones. Figure 2-2 shows the relationship between a node and other network components.

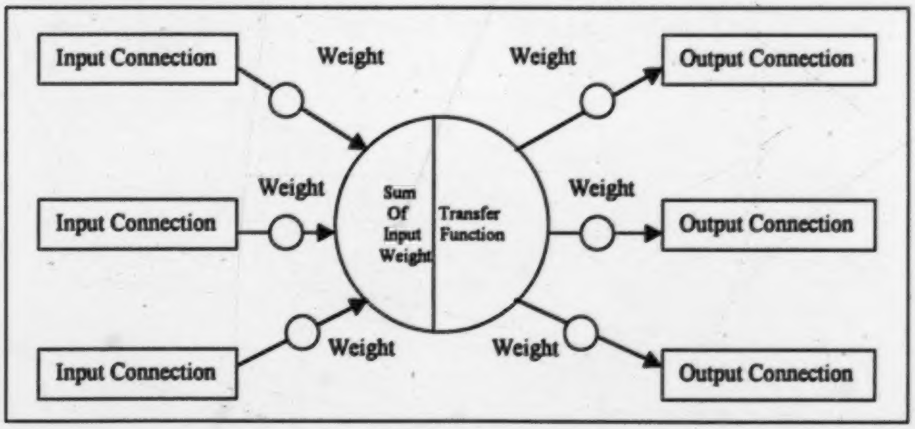

Input Connection
Weight
Weight
Output Connection
Weight
Sum
Of
Input
Weight
Transfer
Function
Weight
Input Connection
Output Connection
Weight
Weight
Input Connection
Output Connection

Figure 2-2 A Node – The Processing Element

Nodes are arranged on layers where every node in one layer is connected to every node in the next layer. The connections between each layer are assigned weights that adjust the importance of an input as it is passed to the next node. The network learns by changing the weights in response to target outputs so as to determine which combinations of inputs are most important. Figure 2-3 shows the basic components of a neural network.

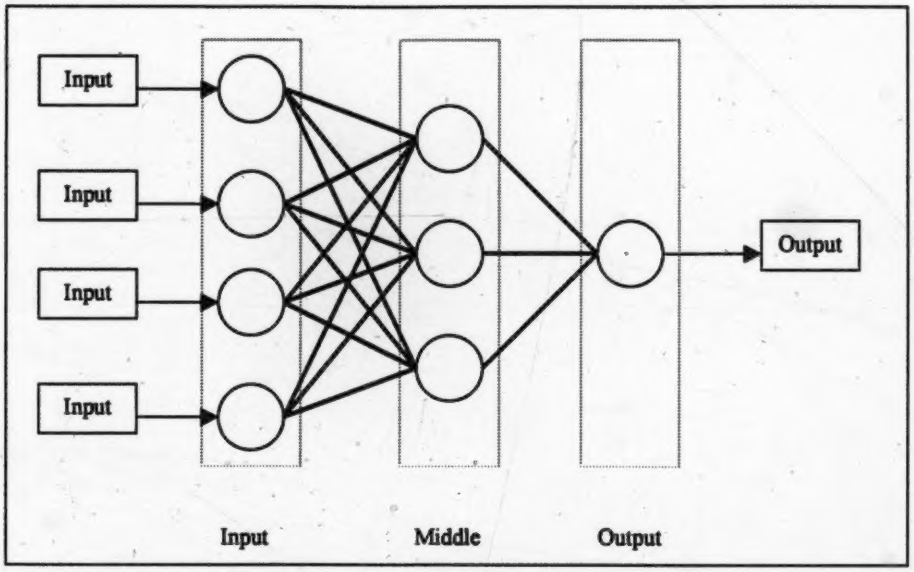
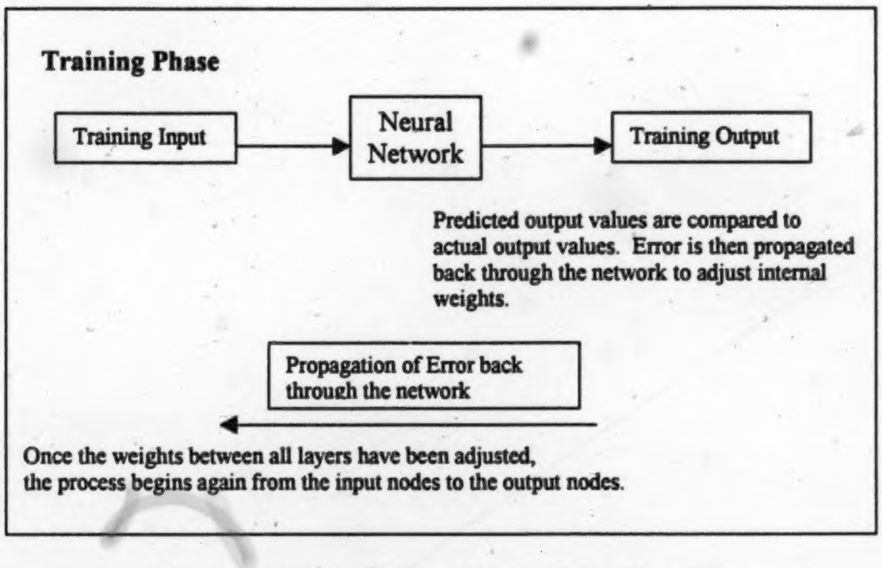


Figure 2-3 Neural Network Structure

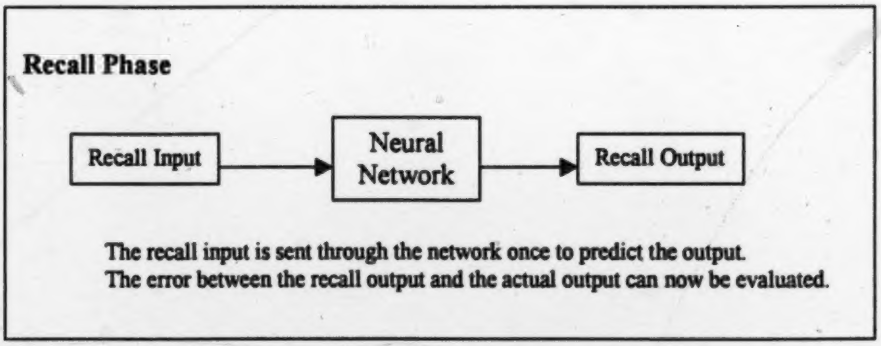
Just like the human brain, neural networks apply knowledge from past experiences to new problems. Neural networks acquire this knowledge by training on a set of data. After this training phase, testing it with data the network has not seen before can validate the model. After validation, the network can be used to predict the type of output the network was trained with. It is important to train the network with a sufficient number of records and use a data set to validate the performance of the network. This data is retained from the training phase for testing only. A general guideline for the minimum amount of data required to train a network is 10(M + N) where M equals the number of inputs and N equals the number of outputs. Another guideline is to have 10 times more training cases than model weights. Figure 2-4 demonstrates the training phase and Figure 2-5 demonstrates the recall phase of neural network development.


Training Phase
Training Input
Neural Network
Training Output
Propagation of Error back through the network

Predicted output values are compared to actual output values. Error is then propagated back through the network to adjust internal weights.

Once the weights between all layers have been adjusted, the process begins again from the input nodes to the output nodes.

Figure 2-4 Training Phase of Network Development


Recall Phase
Recall Input
Neural Network
Recall Output

The recall input is sent through the network once to predict the output. The error between the recall output and the actual output can now be evaluated.

Figure 2-5 Recall Phase of Network Development

Unlike traditional statistical methods, neural networks do not require assumptions about model form. Neural networks learn the patterns in the data, whereas statistical analysis assumes a model form to the data and then tests to see if the data fits the assumed structure. Statistical analysis requires an assumption about which form (i.e. linearity) characterizes relationships between variables. Neural networks are more tolerant of imperfect data or variance from a known model. Neural networks perform better than traditional statistical methods when the model form is unknown or nonlinear, or when the problem is complex with several interrelated relationships. Neural networks have the capacity to learn rapidly and change quickly as data and outcomes change. This feature of neural networks allows for excellent adaptation to constantly changing information.

There are two main learning methods for neural networks: supervised and unsupervised. Supervised learning is used as a prediction tool. It requires historical data with input and output parameters to train the model. With this method, the network can compare the predicted results to the actual results and adjust the model accordingly. The adjustment to the model is done by feeding the error between the predicted and actual output back through the network. This method of correcting the error working from the output end towards the input is called back-propagation. Unsupervised learning is more effective for describing data rather than predicting it. As part of the training process, the neural network is not shown any outputs from which to develop errors. This type of learning groups the data into similar categories and requires no initial assumptions about what constitutes a group or how many groups there should be. This method of training eliminates bias from the factors that should be the most important.

Limitations in neural computing include difficulty in explaining the model in a useful way, and explaining why it predicts the way it does. It is also difficult to extract rules from neural networks. Furthermore, one must spend time understanding the problem or the desired predicted outcome. In order for the network to produce good results, the data used to train the network must be appropriate and representative of the problem being solved.

In conclusion, neural networks are powerful predictive tools for handling large amounts of complex data and recognizing patterns. Neural networks readily adapt to changing information and new data. Neural network performance can be as good as classical statistical modelling and is better for some problems. Although neural networks are computational intensive, they can be solved within reasonable time using today's personal computers.

2.6 Neural Network Applications in Construction

Neural networks are increasingly investigated for use in the construction industry. A number of researchers have discussed neural networks and the numerous applications for them in civil engineering and construction (Moselhi, Hegazy, and Fazio, 1990), (Flood and Kartam, 1994), and (Garrett, 1992). The primary use of artificial intelligence in construction has been in the form of expert systems. The faults of expert systems lie in their lack of ability to learn by themselves, generalize solutions, and adequately respond to incomplete or previously unseen data (Moselhi, Hegazy, and Fazio, 1990). Expert systems involve deep reasoning about the problem elements, whereas neural networks involve pattern recognition, which is more characteristic of the majority of construction problems. Neural networks are seen as a supplement or in some cases a replacement of expert systems.

Some advantages of neural networks over other artificial intelligence techniques are:

1. They are suited for pattern recognition tasks where a large number of attributes must be considered in parallel.
2. They learn by example and model many example patterns and associations.
3. Once developed they produce fast responses.
4. They have distributed memory. The individual connection weights are the memory of the network.
5. They have associative memory. The network responds in an interpolative way to incomplete or previously unseen data.
6. They are fault tolerant. Since the memory is distributed, small failures in the network will only have a slight effect on the overall performance of the network.
7. They require only slight amounts of storage memory since there is only one set of network weights that are capable of representing a large space of stored patterns.

Moselhi et al. (1990) presented a neural network for determining the optimum bid markup. A feed-forward back-propagation three-layered network was used. The input parameters were: the number of typical competitors, the mean of the distribution of the ratio of the competitor's bid prices to the contractor's estimated cost in previous encounters, and the standard deviation of the previously described distribution. Three bidding strategies were used to determine the optimal bid markup for the network. The outputs of the network were three optimal bid markups for each of the different strategies.

Murtaza and Fisher (1994) presented a neural network approach to modular construction decision making. The relevant factors were classified into the following five groups: plant location, labour conditions, environmental and organizational factors, plant characteristics, and project risks. Each of these five major group factors is represented by a two layer neural network. The second layer of each of these networks acts as input into the third layer, which integrates the networks from the five major groups. A multi-layered classification network was utilized and proved to be accurate with both new and incomplete data.

Sawhney et al. (1993) presented a neural network model for accurate forecasting of construction cost escalation. Cost escalation of labour and materials is a viable factor when preparing a bid, especially in long-term projects. A recursive neural network was developed to forecast the prefabricated wooden buildings industry index (PWBII). The neural network was composed of three layers with a recursive portion to model time dependent processes. The input parameters to the network were the month and year, and the output was the price index. The input into the recursive portion included input from the current value of the index and the values from previous iterations. This recursive portion would account for past monthly index values influencing future predicted values from the network. This neural network has the ability of continuously learning and improving its performance based on further training with additional data.

Al-Tabtabai et al. (1996) developed a neural network to identify the variances in quantity of any particular construction work package. Work packages are basically manageable units of a project, defined for the purpose of efficient management and control during construction. The developed network would be used by a project manager to estimate the variance from the estimate quantities gathered during the planning stages of the project. The network was a feed-forward three-layer back-propagation model with six input parameters. These six parameters were: volume of rework, volume of waste and scrap, scope changes, quality of the quantity estimates, percentage of work completed for the work package, and the past and present trend in quantity variance. The output of the network was a ratio of the variance in quantity as compared to the current level of variance. The developed neural network is one component of a construction project control system being developed by the authors. It will include schedule and cost control modules that will be integrated within current developed project management systems.

Savin et al. (1996) developed a neural network model for construction resource leveling. The neural network model is comprised of two main blocks. The first block consists of a discrete-time Hopfield neural network, which is a single-layer feedback network with complete interconnections. The second block is a control block for the adjustment of Lagrange multipliers in the augmented Lagrangian multiplier optimization, and for the computation of the new set of weights for the neural network block. Verification of the model on small sized projects was completed successfully. Difficulty with selecting initial Lagrange parameters for larger projects was encountered with no apparent solutions to the problem. This has led the researchers to suggest that additional investigation is required to apply the developed neural network to medium and large-sized projects.

Chao and Skibniewski (1993) presented an approach to estimate the productivity of a common excavation-hauling operation. Utilizing neural networks proved to be beneficial in mapping the complex and nonlinear attributes of construction productivity. The model contained four input factors and one output factor in modeling an excavator's cycle time under various conditions. Data for the network was obtained with simulation by means of a small-scale robotic excavator.

Wales and AbouRizk (1996) developed a neural network application for estimating construction labour productivity. The factors included in this model consisted of environmental conditions such as temperature, relative humidity, precipitation, and wind speed. These environmental processes were considered to be random events within the simulation model. The random events were given to the neural network to predict the daily productivity, which was then used in the simulation process to generate construction schedules.

AbouRizk et al. (1996) developed a neural network for estimating trenching productivity in pipeline operations. A feed-forward back-propagation model was used. The five input factors were: hours worked per day, percent of project complete, temperature, type of equipment used, and weather severity. The output of the network was the predicted rate of pipeline trenching productivity.

Portas (1996) developed a neural network application to aid in the estimation of labour productivity for concrete formwork. A feed-forward back-propagation network was used that contained fifty-five input nodes and thirteen output nodes. A total of 30 factors that affect formwork productivity were input into the model through the fifty-five input nodes. The thirteen output nodes represent a binary output pattern corresponding to subset ranges of productivity values. Ranges of productivity values were given by the network instead of a point estimate to account for uncertainty and to reduce errors from poor training data. From this research, knowledge about which factors affect labour productivity was gained and a predictive tool for estimating formwork productivity on future projects was developed.

2.7 Discussion of Belief Networks

Belief networks, also known as influence diagrams, causality diagrams or Bayesian networks, were first developed in the 1970's. Belief networks can be thought of as a way to model a situation in which causality plays a role, but where an understanding of what is actually happening is fuzzy, so things must be described probabilistically. Belief networks are directed, acyclic graphs (DAG), made up of arcs and nodes; they utilize Bayes' Theorem and the concepts of conditional probability. A simple belief network as adopted from Charniak (1991) is shown in Figure 2-6. This network models the situation where a father wants to determine whether his family is home or not before he enters the home. The factors which are used to model the situation are whether the family is home or not, the outside light is on or not, the dog is out or not, the dog has a bowel problem or not, and if the dog barks or not. This network will be further discussed later on in this section.

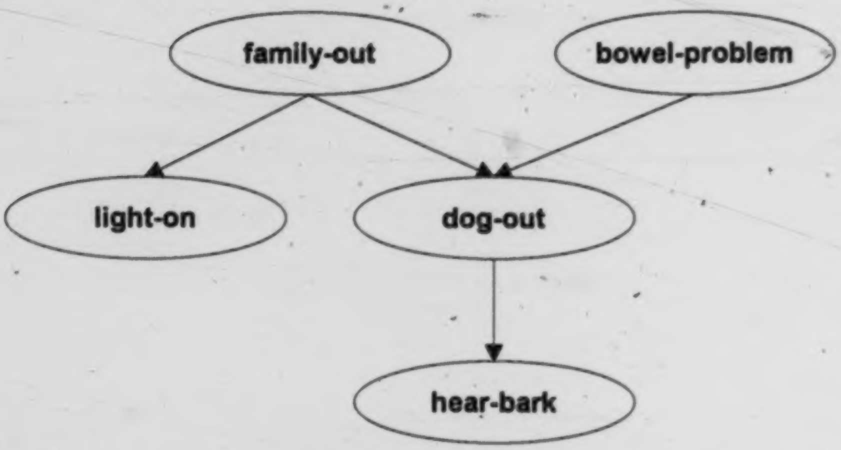


Figure 2-6 A Simple Belief Network

The nodes represent the variables of the domain, and the arcs represent the dependence between the nodes. The term 'directed' refers to the fact that the arcs have an explicit direction and are represented by arrows showing the direction. 'Acyclic' means that the arrows may not form a directed cycle or loop in the network (i.e., the path cannot be circular when the path of arrows is considered). The node at the beginning of the arc is called the parent, whereas the node at the arrow end of the arc is known as the child node. The parent is assumed to affect the states of the child. In Figure 2-6 a parent node could be 'family-out' with the child nodes being 'light-on' and 'dog-out'. Nodes not directly joined by arcs are either independent, or may be evaluated as being conditionally independent. The belief network can predict causal outcomes differently based on the evidence or observed conditions entered into a constructed network. One might be tempted to think that the probabilities of the nodes are changing, when actually the conditional probability is changing as a result of the changing evidence. One important feature of belief networks is the simplification of probability calculations as compared to traditional probabilistic models. In traditional models the elements are completely interconnected, whereas in belief networks they are only connected to factors that can affect them. The number of probabilities required is much lower due to the built-in independence assumptions within belief networks. This is evident in Figure 2-7, where each network has the same number of nodes, but one has more connections between nodes than the other. Network 1 represents a more traditional method of connecting nodes where each node is connected to all other nodes. Network 2 represents how nodes are connected in a belief network where there is only one connection between each node. Assuming that each node is binary (i.e., has only two states), the number of probabilities that must be evaluated for the state of E in Network 1 is $2^4 = 16$. The node has two possible states and four parents. However, in Network 2 the number of probabilities required is $2^1 = 2$. By reducing the number of connections between the nodes, one improves the efficiency of the model. Also by structuring the network effectively, the number of probabilities required to represent the real system is reduced, resulting in a more efficient model. Network 2 has an exact solution, whereas there is no polynomial time algorithm to determine the solution of Network 1 Charniak (1991). Research continues to improve the techniques for developing more efficient algorithms to solve these types of networks.

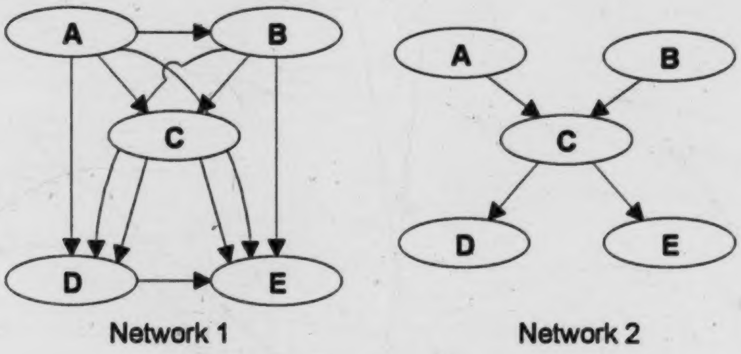


Figure 2-7 Network Structure Comparison

Independence can occur when two variables can cause the same result, but are in no other way connected. For example, either the family being out or the dog having a bowel problem can cause the dog to be put out. The network shows that the family being out has no direct relationship with the dog having a bowel problem. However, these two variables can be conditionally dependent on each other if evidence of the result is entered into the network during evaluation. The built-in independence assumptions are input into the network with the guidance of an expert. This expert subjectively assesses the probabilities that are necessary for the evaluation process in a belief network. One study of doctors' assessments (Spiegelhalter et al., 1989) showed that the assessment probabilities required for a belief network were indeed very close to the numbers that were subsequently collected. One problem cited in this research was that the doctors were typically too quick to give assess as a zero probability of occurrence.

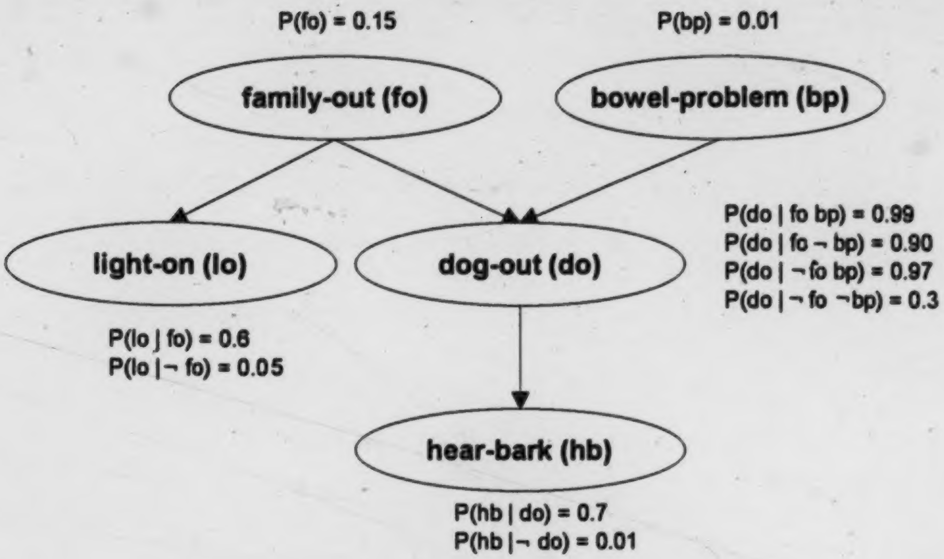

P(fo) = 0.15
P(bp) = 0.01
family-out (fo)
bowel-problem (bp)
light-on (lo)
dog-out (do)
hear-bark (hb)
P(do | fo bp) = 0.99
P(do | fo ¬ bp) = 0.90
P(do | ¬ fo bp) = 0.97
P(do | ¬ fo ¬bp) = 0.3
P(lo | fo) = 0.6
P(lo | ¬ fo) = 0.05
P(hb | do) = 0.7
P(hb | ¬ do) = 0.01

Figure 2-8 A Belief Network for the Family-out Problem

Figure 2-8 shows a belief network with the attached conditional prior and posterior probabilities. Prior probabilities are those assigned to nodes with no parents (i.e., nodes at the top of the network). Posterior probabilities are those assigned to child nodes that are conditional to the various combinations of the states of the parents. To define a few of the symbols in a belief network, refer to Figure 2-8. The statement P(lo|fo) = 0.6 means that the probability of light-on = true given that family-out = true is 0.6. The statement P(lo|¬fo) = 0.05 means that the probability of light-on = true given that family-out = false is 0.05. The statement P(do|fo∧bo) = 0.99 means that the probability of dog-out = true given that family-out = true and bowel-problem = true is 0.99. Knowing what the symbols mean we can see that if the family leaves the house, they will turn on the outside light 60 percent of the time, but the light will be turned on even when they do not leave 5 percent of the time (i.e., a guest is expected). Now giving evidence to the network will involve conditional probabilities to be calculated by the network. If it is observed that the light is on (i.e., light-on = true) but the dog is not heard barking (i.e., hear-bark = false), the conditional probability that the family-out = true calculates to 0.5. The calculation of conditional probabilities is usually referred to as 'evaluating the belief network'.

Belief networks use Bayes' Theorem, as shown in Equation 1,which follows from the basic conditional probability relationship $P(A \wedge B) = P(B|A)*P(A) = PA|B)*P(B)$. Bayes' Theorem may also be used to analyze multiple influences in the form of Equation 2, where the denominator is the expansion of the denominator in Equation 1, and is the unconditioned P(A = true).

Equation 1

$$P(B \mid A) = \frac{P(A \mid B) * P(B)}{P(A)}$$

Equation 2

$$P(B_i \mid A) = \frac{P(A \mid B_i) * P(B_i)}{\sum_{k=1}^{n} P(A \mid B_k) * P(B_k)}$$

For an illustration of how a belief network works refer to Figure 2-8. This simple network is designed to evaluate whether a family is home or not before the home is entered. All of the variables in this network are binary (i.e., either true or false).

Now consider the situation where the person approaching the home hears the dog bark and observes that the light is not on. Given this evidence, the person now wants to know if the family is out or not given this evidence. The problem statement is: $P(fo|hb \wedge \neg lo)$ where fo represents the true state of the node family-out, hb represents the true state of the node hear-bark, and $\neg lo$ represents the false state of the node light-on. With all information contained in the network relying upon conditioning on the parent, the problem statement must be manipulated until the required information may be read directly from the network. Using Equation 1, the problem statement is rearranged so that it is conditioning on a parent.

$$P(fo \mid hb \wedge \neg lo) = \frac{P(hb \wedge lo \mid fo) * P(fo)}{P(hb \wedge \neg lo)}$$

P(fo) may be read directly from the network, but the other two elements require further analysis. With fo assumed to be known, the two variables hb and lo are D-separated and are therefore independent. To explain D-separation, also known as direction-dependent separation, consider nodes A and E in Network 2 of Figure 2-7. The nodes are obviously connected and are therefore dependent upon one another. However, if a node between them is known and there is no other path between them that is not blocked by any given node, then the two become D-separated, or independent of each other. So the equation may be restated as:

$$P(hb \wedge \neg lo \mid fo) = P(hb \mid fo) * P(\neg lo \mid fo)$$

To evaluate P(hb|fo), the probability of hb must be conditioned on all of the parents. So the node is evaluated for the given information (i.e., P(fo = true), and on all conditions of the remaining parents.

$$P(hb \mid fo) = P(hb \mid fo \wedge do) * P(do \mid fo) + P(hb \mid fo \wedge \neg do) * P(\neg do \mid fo)$$

In the expression P(hb|fo∧do), hb and fo have become D-separated by do, and now the probability of hb now only depends upon do. The term may now be expressed as P(hb|do), leaving P(do|fo) to be evaluated with all combinations of its parents. Also note that P(¬do|fo) = 1-P(do|fo).

$$P(do \mid fo) = P(do \mid fo \wedge bp) * P(bp) + P(do \mid fo \wedge \neg bp) * P(\neg bp)$$

The numerator of the problem statement is now in a form where the information may be read directly from the network. The denominator may be restated as:

$$P(hb \wedge \neg lo) = P(\neg lo \mid hb) * P(hb)$$

Because P(¬lo|hb) = 1- P(lo|hb), the evaluation of P(¬lo|hb) may be simplified to the following:

$$1 - P(lo \mid hb) = 1 - (P(lo \mid hb \wedge fo) * P(fo \mid hb) + P(lo \mid hb \wedge \neg fo) * P(\neg fo \mid hb))$$

$$P(fo \mid hb) = \frac{P(hb \mid fo) * P(fo)}{P(hb)}$$

and

Now lo and hb have been D-separated by fo and therefore reducing the term P(lo|hb∧fo) to P(lo|fo) which can be read directly from the network. With the value of P(hb|fo) already being evaluated, all but P(hb) can be read from the network. So hb is evaluated by being conditioned on all combinations of the parents.

$$P(hb) = P(hb \mid do) * P(do) + P(hb \mid \neg do) * P(\neg do)$$

and

$$\begin{aligned} P(do) = {} & P(do \mid bp \wedge fo) * P(bp) * P(fo) + P(do \mid bp \wedge \neg fo) * P(bp) * P(\neg fo) \\ & + P(do \mid \neg bp \wedge fo) * P(\neg bp) * P(fo) + P(do \mid \neg bp \wedge \neg fo) * P(\neg bp) * P(\neg fo) \end{aligned}$$

Now the network provides all of the information required for evaluating the problem statement. Working through the above equations in the reverse order we find:

$$P(do) = (0.99 * 0.01 * 0.15) + (0.97 * 0.01 * 0.85) + (0.9 * 0.99 * 0.15) + (0.3 * 0.99 * 0.85)$$
$$= 0.3583$$
$$P(hb) = (0.7 * 0.3583) + (0.01 * (1 - 0.3683) = 0.2831$$

$$P(do \mid fo) = P(do \mid fo \wedge bp) * P(bp) + P(do \mid fo \wedge \neg bp) * P(\neg bp)$$
$$= (0.99 * 0.01) + (0.9 * 0.99) = 0.9009$$
$$P(hb \mid fo) = P(hb \mid do) * P(do \mid fo) + P(hb \mid \neg do) * P(\neg do \mid fo)$$
$$= (0.7 * 0.9009) + (0.01 * (1 - 0.9009)) = 0.6316$$
$$P(hb \wedge \neg lo \mid fo) = P(hb \mid fo) * P(\neg lo \mid fo) = P(hb \mid fo) * (1 - P(lo \mid fo)$$
$$= 0.6316 * (1 - 0.6) = 0.2526$$

$$P(fo \mid hb) = P(hb \mid fo) * P(fo) / P(hb) = (0.6316 * 0.15) / 0.2831 = 0.3346$$
$$P(\neg lo \mid hb) = 1 - (P(lo \mid hb \wedge fo) * P(fo \mid hb) + P(lo \mid hb \wedge \neg fo) * P(\neg fo \mid hb))$$
$$= 1 - ((0.6 * 0.3346) + (0.5 * 0.6653)) = 0.7660$$
$$P(hb \wedge \neg lo) = P(\neg lo \mid hb) * P(hb) = (0.7660 * 0.2831) = 0.1771$$

$$P(fo \mid hb \wedge \neg lo) = P(hb \wedge \neg lo \mid fo) * P(fo) / P(hb \wedge \neg lo) = (0.2526 * 0.15) / 0.1771 = 0.21$$

The homeowner could therefore conclude with 21% confidence that the family is out, or with 79% confidence that the family is in. Evaluating the situation where the light is on and the dog is not heard barking gives a result of 50% confidence that the family is out.

In general, belief networks can be used to predict what will happen (i.e., the family goes out, the dog goes out) or to infer causes from observed effects (i.e., if the light is on and the dog is out, then the family is probably out). Evaluation of a belief network basically involves the computation of every node's belief (i.e., its conditional probability) given the evidence that has been observed so far. A conditional probability is a probability or likelihood of a variable that is dependent on the value of another variable. For example, given that the grass is wet, the cause could either be that it just rained or that the sprinkler system was just on. The likelihood that it has rained is differs depending on whether one knows if there was a rainstorm in the area. If it is known that there was a rainstorm that had passed by recently, then the likelihood that the sprinkler system was the source of precipitation is significantly less than if it was known that a rainstorm was not in the area.

Belief networks are very flexible in how input is accepted and output is provided. They allow for variables to be either input or output without redesigning the system. This characteristic is not present in many other forms of artificial intelligence. Rule-based expert systems are based on a number of 'expert' rules and do not require large amounts of data, much like belief networks. The advantage of using belief networks is that rule-based expert systems permit evidence to enter the system only at specified points; they must be restructured to allow for intermediate points to accept evidence. Expert systems determine the causes given only the evidence and require an entire new set of rules to work in the opposite direction. Another form of artificial intelligence, known as neural networks, has a more rigid structure than belief networks or expert systems. If any of the variables are changed, then an entirely new network must be created to model the data. One major advantage of belief networks is that they have the ability to both accept evidence at any point in the system and provide output at any point in the system (Henrion et al., 1991). Belief networks are mainly used for diagnostics, also called diagnostic inference. This is where evidence of the symptoms are supplied to the network and the network determines the likely cause for the given symptoms. However, belief networks can also be used to provide information about the symptoms and the network can determine the likely result. This is called 'causal inference' and can be done utilizing the same network structure. Belief networks can also be used for what is called 'intercausal inference' (Henrion et al., 1991). Here, the belief of each node is determined with the entry of additional evidence. New evidence is entered at any point in the network, and the likelihood of the remaining variables is determined and compared to the belief values evaluated before the new evidence was given to the network.

To have the greatest affect on the conditional probabilities when using a belief network for diagnostic purposes, the evidence nodes should be at the bottom of the belief network. The model developed in this research will follow this guideline and the evidence nodes will be found at the bottom of the network. When any evidence is found it will be entered at these points and the network will be evaluated. The reason for this is that when entering evidence into a node in the middle of the network, one does not include any of the node's dependent nodes (i.e. nodes below) to give us the probable cause at the top of the network.

The software used in this research is called Microsoft® Bayes Networks (MSBN™). MSBN (Microsoft Corporation, 1996) integrates with both C and Visual Basic programming code and is therefore compatible with the developed CRUISER program. This will allow for possible integration between the two software packages for practical purposes.

2.8 Belief Network Applications in Construction

Some artificial intelligence researchers like Charniak (1991) assert that, despite the importance of these networks, the ideas and techniques have not progressed beyond the research community that developed them. This is most likely due to the fact that the ideas and techniques are not very easy to understand. The main problem with incorporating expert opinion into a belief network is the general lack of understanding of probability theory. Some applications for belief networks have been found in other fields like environmental engineering (Chong and Walley, 1996), medicine, and software development (Heckerman and Wellman, 1995). One application of belief networks for the construction industry has recently occurred in the area of earthmoving operations. In this research, a belief network was developed to assist the user of an earthmoving simulation model in determining what parameters to change in order to optimize either the production or the construction schedule (McCabe, 1997). The belief network was designed to perform a number of iterations with the earthmoving model, and provide the results in an output file. The earthmoving model was to be designed by the user with Visual SLAM® and AweSim® simulation software. In the output file the user can see how each iteration change made by the belief network further optimized the model. The model can be optimized by one of the two methods chosen by the user. Although belief networks have not been utilized very much in the construction industry, it is the desire of the researcher to expand their use through this research.

3.0 CRUISER Validation

3.1 Introduction

In order to determine where improvements can be made, a validation process for a simulation model must be undertaken. Data from the real situation is usually the best place to start. Previous research has been done to validate the CRUISER simulation model with actual aggregate plant data (Chehayeb, 1996). Data was collected from a total of three different aggregate plants. One important finding in this research is that the output gradation is quite dependent on the gradation of the input feed. One modification to the CRUISER program was to implement a stochastic modeling process so that the user would supply more than one raw feed gradation. The model then runs a specified number of simulations and the final product gradation is supplied with production statistics such as the mean, standard deviation, percentile values, and confidence intervals. The results show that the final product variations depend on variations in inputs. Model accuracy was best with the largest plant of the three, which has a total of two crushers and six screens. The CRUISER gradation varied at this plant from -12% to +8% on the cumulative percent-passing curve. At the second plant 20mm and 50mm products were made. The CRUISER gradation varied from -30% to +5% on the cumulative percent-passing curve. The third plant produced a 40mm product and the CRUISER varied from -25% to 0% on the cumulative percent-passing curve. This researcher attributes inaccuracy of CRUISER results to the inconsistency of production from small, mobile aggregate plants. The problem with this reasoning is that relatively similar accuracy was obtained from plants with more crushers and screens. The accuracy of the model should be independent of the size of a plant if the crushing and screening processes are properly simulated by CRUISER. Results from this research consisted mostly of improvements to the user interface and graphical displays of output data within CRUISER. Chehayeb (1996) recommended collecting data on the input and output of every screen and crusher in order to perform a more detailed validation analysis of the CRUISER model. Another suggestion was to use neural networks to simulate the crushing and screening processes within CRUISER.

For data collection in this research, a number of site visits were made to the Lafarge crushing plant just north of Villineuve, Alberta. The data collection took approximately one month starting on October 1, 1996. A total of seven samples were taken over the course of the month. Four of these samples contain aggregate gradations for all locations of the plant. Samples were collected and tested onsite before and after each screen and crusher. The other three samples contain aggregate gradations for the raw pile, the waste sand pile, and the product pile. This is because after the fourth sample was obtained, the plant could not be shut down in mid-stream for data collection purposes. From then on, only samples were taken from the input, and output of the sand and product streams. The main purpose here was to see if actual data obtained in the field was comparable to what CRUISER would evaluate. This procedure was done to test how accurately CRUISER would predict versus the 'real world' aggregate processing.

3.2 Data Collection

3.2.1 Procedure

A Data Collection Handbook was made for data collection purposes when site visits occurred. It consisted of a number of pages for information on site layout, crusher information, screen information, conveyor information, and site reports. Site reports contained the pertinent site visit information, such as changes in operations, loader analysis, conveyor analysis, production rate analysis, and the completed sieve tests. An example of the Data Collection Handbook is shown in Appendix C.

After the first site visit to the plant it was decided that additional information might be obtained by taking the samples off of the conveyors in a particular way. This would allow for obtaining production rates for each aggregate stream sampled. These production rates could then be compared with the rates obtained from the CRUISER program after analysis. The production rate gained from this process would be used as a check on the values of raw feed and final product. For the raw product rate, a loader analysis was done in which the number of loads and the approximate load volume over a given time were recorded, then a production rate was calculated. The time over which this analysis was done for each site visit was about 20 minutes. For the final product, a weigh scale was present to record the instantaneous production rate for the product. This scale was monitored for a period of about 20 minutes; recordings were taken every minute on each site visit. The density for each aggregate sample from each conveyor was also calculated. This was done using the samples obtained for the sieve analysis and a volumetric pail for which a volume was known. The density of the aggregate was determined using CSA Test Method A23.2-10A. The sieve analysis of the fine and coarse aggregate was determined using CSA Test Method A23.2-2A.

3.2.2 Samples

A total of four complete sample sets have sieve analysis data for all components of the aggregate plant. Three of these samples were taken when Asphalt Concrete Overlay (ACO) was being made. The other sample was taken when Asphalt Concrete Residential (ACR) was being made.

An additional three samples were obtained with sampling from the raw, waste, and product piles. The reduced amount of sampling was due to the constraints of limited plant shutdowns. These samples were obtained during normal plant operations with no plant shutdown required. These additional samples would be used in evaluating the plant as a whole instead of the individual components separately. One product type was 20mm Road Crush; the other two were ACR product, which was being made for the City of Edmonton. The main difference between the ACR product for Lafarge or the ACR product for the City of Edmonton is that the ACR product for the City has 6% more sand than the same product type made for Lafarge. An additional difference between ACO and ACR is that ACO contains 80% fractured rock, and ACR contains 75% fractured rock in their respective specifications.

3.2.3 Analysis

Differences between the alternative methods of collecting data for the input production rate are shown in Table 3-1. Table 3-2 shows the comparison of the two output production rate methods. Full details for these tables are included in Appendix D. We can see that the different methods of calculating the tonnes per hour (TPH) for the input (i.e. raw product) and the output (i.e. final product) were similar in results. The greatest difference between any two methods was 18%. Errors were larger between the methods for the output rates. The average error there was 12.2%. This might be due to inaccuracy of the data because of weight scale that was used. The error between the methods on the input end seemed to be quite low, with an average error of 6.6%. These results are quite good considering that the loader analysis method involved quantitative observations regarding loader volumes. Another factor to consider in the accuracy of the TPH calculations obtained from the samples collected is that two individuals collected all of the samples during any one site visit. This was done because the shutdown time of the plant had to be minimized. The fact that the gradations were analyzed by more than one person contributed to inconsistency errors. The time constraint during sample collection is likely to have contributed some error as well.

Table 3-1 Input Production Analysis

Sample Description	Conveyor TPH	Loader Analysis TPH	% Difference
ACO Trial #2	538.2	503	7
ACO Trial #3	528.2	521.2	1.3
ACR Trial #4	598.6	563.7	6.2
RC 20mm Trial #5	677	582.3	16.3
ACR CoE Trial #6	455.4	494.5	- 7.9
ACR CoE Trial #7	522.2	528.4	-1.2
		Average Difference	**6.6 %**

Table 3-2 Output Production Analysis

Sample Description	Conveyor TPH	Weigh Scale Analysis TPH	% Difference
ACO Trial #2	370.5	313.8	18.1
ACO Trial #3	391.2	335.5	16.6
ACR Trial #4	379.7	372.2	2
		Average Difference	**12.2 %**

The method of determining the density of each sample was deemed to be highly relevant and accurate. The densities did not always make sense when one looked at the different products within a given sample. Typically, the more the raw product is crushed, the denser it becomes. This is not necessarily observed when one looks at a specific sample. However, when analyzing between samples with the same final product being made, the numbers between individual samples are quite close. The density for the screened pit run (i.e., the second sampling location) increased; this is because the majority of sand was removed by this point, leaving more rock in the sample and creating a corresponding higher density. Any small errors within a sample (i.e., before and after a crusher) are likely due to an absence of compaction of the samples when weighed for volumetric purposes. It is to be noted, though, that the researcher was looking for the "as-is" state density at the time; this explains why the density values for the samples from a given test as a whole might not appear to make sense.

3.3 Plant Characteristics

The model of the crushing plant is as shown below in Figure 3-1.

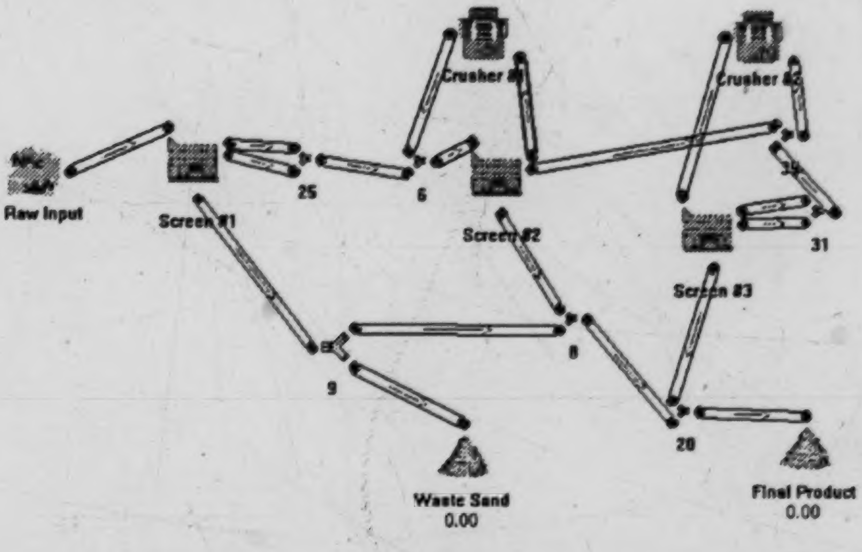

Crusher #1
Crusher #2
Raw Input
Screen #1
25
6
Screen #2
34
31
Screen #3
9
8
20
Waste Sand
0.00
Final Product
0.00

Figure 3-1: Site Model (as depicted using CRUISER)

Tables 3-3, 3-4, and 3-5 outline the screen information for Screens 1 through 3. The settings of the components within CRUISER were set as follows:

Screen #1

Deck Size:	5' x 14'
Condition:	Dry quarried material, 4% or less moisture; crushed rock
Incline Factor:	Horizontal, Normal Amplitude Stroke

Table 3-3 Screen #1 - Screen Information

Deck #	Split #	Opening (inches)	Slot Length/Width	Open Area Factor
1	1	1.250	1:1 (Square)	50% - Standard Wire
	2	0.875	1:1 (Square)	50% - Standard Wire
	3	0.875	1:1 (Square)	50% - Standard Wire
	4	1.250	1:1 (Square)	50% - Standard Wire
2	1	0.375	More Than 6:1	50% - Standard Wire
	2	0.375	1:1 (Square)	50% - Standard Wire

Screen #2

Deck Size:	6' x 20'
Condition:	Dry quarried material, 4% or less moisture; crushed rock
Incline Factor:	20 Degrees

Table 3-4 Screen #2 - Screen Information

Deck #	Split #	Opening (inches)	Slot Length/Width	Open Area Factor
1	1	1.250	1:1 (Square)	50% - Standard Wire
2	1	0.438	1:1 (Square)	50% - Standard Wire

Screen #3

Deck Size:	6' x 20'
Condition:	Dry quarried material, 4% or less moisture; crushed rock
Incline Factor:	20 Degrees

Table 3-5 Screen #3 - Screen Information

Deck #	Split #	Opening (inches)	Slot Length/Width	Open Area Factor
1	1	0.750	1:1 (Square)	70% - Very Light Wire
2	1	0.563	1:1 (Square)	70% - Very Light Wire

	Crusher #1	Crusher #2
Type:	Cone	Cone
Size:	54"	36" (Actually 66")
Setting:	1.000"	0.4375"
Suggested Capacity:	181 TPH	35 TPH
Set Capacity:	400 TPH	250 TPH

3.4 Product Sampling and Results

3.4.1 ACO Product Type

Three samples were collected and tested individually by way of the deterministic feature in CRUISER. The material weight was specified as 113 lbs per cubic foot. This was an average from the data collection process for the ACO product types. The most likely feed rate was set at 530 TPH.

The stochastic sampling option within CRUISER was attempted with the three ACO samples. For the stochastic process a low of 510 TPH, a mode of 530TPH, and a high of 540 TPH was used. This was estimated based on the input rates determined from the data collection process. Another important note is that the percent flow of sand to the final product was set at 6% for the ACO product type. The resulting output product gradation carries upper and lower bounds along with it. These upper and lower bounds are found to vary up to ±2%, which does not improve the gradation results enough to bring them within the gradation specifications. It was found that with similar raw input gradations better results were achieved. With inputs that are more varied, the results become skewed towards the coarser or finer input results, depending on whichever input type is greater in number. Varying the high and low feed rates to achieve a greater variability of input rate does not affect the variance of the upper and lower bounds any more than ± 2%, as before. In conclusion, it is best to have several input gradations that reflect the actual conditions most closely. However, the stochastic sampling feature within CRUISER still will not improve the results enough to meet specifications, which is the desired goal for the program user.

3.4.1.1 Results

A graph presenting the average results for the three samples along with the product specifications is shown in Figure 3-2.

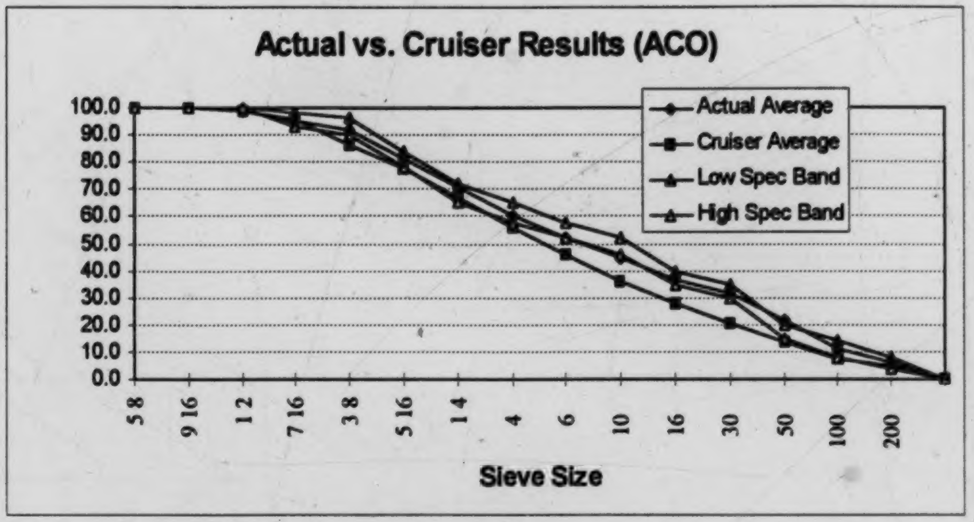


Figure 3-2 Actual vs CRUISER Results for ACO Product Type

Additional results presented in graphical form for the ACO product type are located in Appendix E. A few general comments can be made for this product type: The raw input gradation varies substantially, especially on the larger sieve sizes. The actual product gradation varies 8% at the most among the samples. The product gradation obtained from CRUISER is fairly consistent and is less variable than the actual product gradation. When evaluating all three samples at once, we can see that the three actual samples reflect a finer product than what CRUISER predicts throughout most sieve sizes. The average product gradation obtained from CRUISER is approximately 3 to 5% lower on the cumulative percent passing graph for the larger sieve sizes. CRUISER is approximately 5 to 11% lower on the smaller sieve sizes. This indicates that CRUISER results are coarser than the actual product. CRUISER is up to 1% higher on the cumulative percent passing graph, revealing that CRUISER results are just slightly finer on one larger sieve size only. The actual product gradation meets the product specifications, whereas the product predicted by CRUISER does not meet the product specifications.

3.4.2 ACR Product Type

Three samples of this product type were collected and tested individually by way of the deterministic feature in CRUISER. The material weight was specified as 109 lbs per cubic foot. This was an average obtained from the data collection process for the ACO product types. The most likely feed rate was set at 550 TPH. Out of the three samples, one was a 'Lafarge' ACR while the other two were 'City of Edmonton' ACR products. The main differences between the two are in the specifications and the material handling process. The 'City of Edmonton' ACR has 6% more sand than the 'Lafarge' ACR product. The product was collected off of the continuous conveyor into a small collecting hopper, which was used to store the product temporarily before it was loaded onto dump trucks. This was a requirement in the specifications from the City and was enforced mostly to prevent segregation of the final product. Usually, the product is stockpiled using large conveyors which create large product stockpiles, thus increasing the amount of segregation.

The stochastic sampling option within CRUISER was attempted with the three ACR samples. For the stochastic process, a low of 500 TPH, a mode of 550TPH, and a high of 600 TPH was used. This was estimated based on the input rates determined from the data collection process. Another important note is that the percent flow of sand to the final product was set at 12% for the ACR product type. The resulting output product gradation carries upper and lower bounds along with it. These upper and lower bounds are found to vary up to ± 2%, which does not improve the gradation results enough to bring them within the gradation specifications. It was found that more comparable raw input gradations usually increased the result quality. The presence of some coarse and some finer inputs will skew the results depending on whichever input types are greater in number than the other. Varying the high and low feed rates to achieve a greater variability does not affect the variance of the upper and lower bounds any more than ± 2%, as before. In conclusion, it is best to have numerous inputs which reflect the actual conditions most closely. However, the stochastic sampling results will still not improve the results to within specifications, i.e. the desired goal.

3.4.2.1 Results

A graph presenting the average results for the three samples along with the product specifications are shown in Figure 3-3.

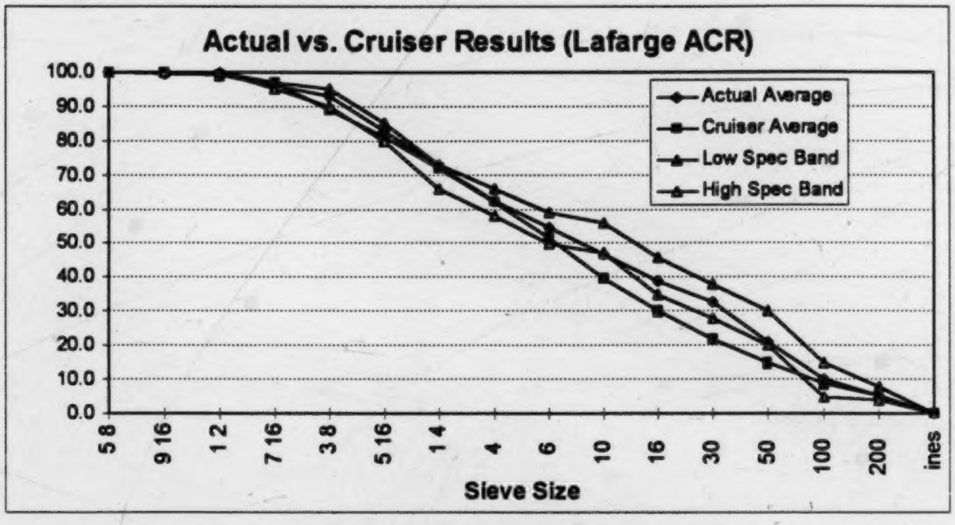


Figure 3-3 Actual vs CRUISER Results For ACR Product Type

Additional results presented in graphical form for the ACR product types are located in Appendix F. A few general comments about this product type can be made: The raw input gradation varies substantially, especially on the larger sieve sizes. One sample in particular seems to deviate from the other two quite significantly. The actual product gradation varies 9% at the most among the samples. The product gradation obtained from CRUISER is fairly consistent for two of the samples, but on the whole is more variable than the actual product gradation. By evaluating all three samples at once it was found that the three actual samples are generally finer than what CRUISER predicts. The actual gradation of the third sample becomes coarser than what CRUISER predicts for a few of the higher sieve sizes. This could be due to sampling or testing error. The average product gradation obtained from CRUISER is about 1 to 5% lower on the cumulative percent passing graph for the larger sieve sizes. CRUISER is about 2 to 11% lower on the smaller sieve sizes. This indicates that CRUISER results are coarser than the actual product. CRUISER is up to 2% higher on the cumulative percent-passing graph, indicating that CRUISER results are finer on one larger sieve size only. The actual product gradation meets both product specifications (i.e. Lafarge and City of Edmonton), whereas the product predicted by CRUISER does not meet the specifications of either product entirely.

3.4.3 20mm Road Crush Product Type

Here, only one sample was collected and tested using the deterministic feature within CRUISER. The material weight was specified as 118 lbs per cubic foot. This value was obtained from the data collection process for this product type. The most likely feed rate was set at 677 TPH. No stochastic sampling was attempted since only one sample of this type was obtained.

3.4.3.1 Results

A graph presenting the average results for the three samples along with the product specifications is shown in Figure 3-4.

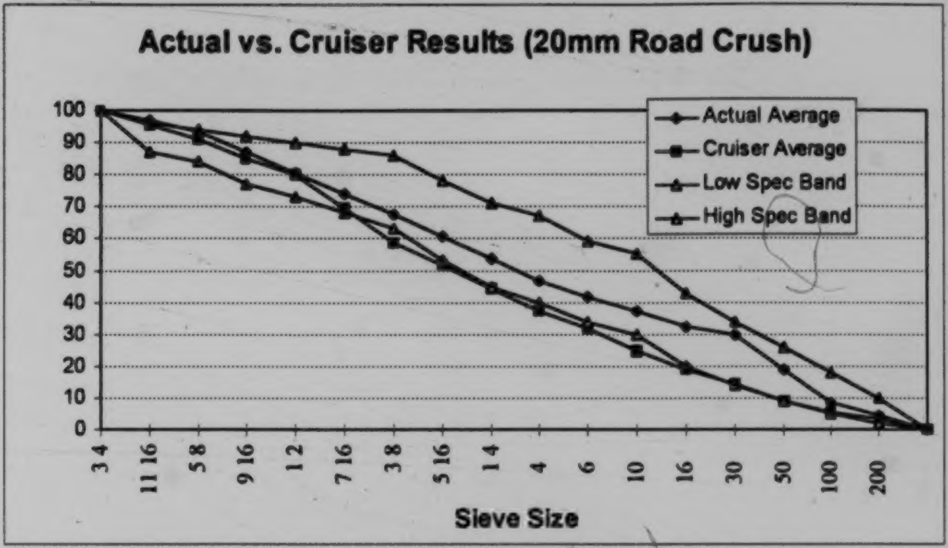


Figure 3-4 Actual vs CRUISER Results For 20mm Road Crush Product Type

Additional results presented in graphical form for the 20mm Road Crush product type are located in Appendix G. A few general comments can be made for this product type: The raw input gradation is characteristic of the other raw input samples. The actual product gradation cannot be compared to other samples of this type since only one sample was obtained. The product gradation obtained from CRUISER cannot be compared to other samples of this type since only one sample was obtained. The product gradation obtained from CRUISER is approximately 2 to 10% lower on the cumulative percent-passing graph for the larger sieve sizes. CRUISER is approximately 2 to 16% lower on the smaller sieve sizes. This indicates that CRUISER results are more coarse on the middle to lower sieve sizes. CRUISER does not predict a finer product on any sieve size for this product type. The actual product gradation meets the product specifications, whereas the product predicted by CRUISER does not meet the product specifications on the middle and lower sieve sizes.

3.4.4 TPH Analysis

Table 3-6 below shows the production in tonnes per hour (TPH) for each of the respective streams for all of the samples. Only four of the seven samples have actual individual component data to compare to CRUISER predictions. Average TPH rates were taken for crushers where the input and output rates should be equal. It can be seen from the table that CRUISER does not predict the tonnage of intermediate streams very accurately, but does indeed predict the final product tonnage within an average of 13.7%. For the two ACO product types the average difference between the actual and the production predicted by CRUISER is 6.1%. The error difference between the actual and predicted production rates is directly related to the gradation of the intermediate streams. The error is largest in the crushing input and output streams. This may be due to the inaccuracy of the crushing analysis within CRUISER.

Table 3-6 Actual and CRUISER Simulated Results

	ACO #1	ACO #2	ACO #3	ACR #1	ACR #2	ACR #3	RC
Pitrun	530	530	530	550	550	550	677
Sand (Cruiser)	140.44	178.55	164.13	55.88	202.18	58.18	110.39
Sand (Sampled)	n/a	220	201	145	n/a	n/a	n/a
Difference	n/a	23%	23%	159%	n/a	n/a	n/a
Screened Pitrun (Cruiser)	380.6	340.05	355.4	483.09	320.25	483.88	559.57
Screened Pitrun (Sampled)	n/a	260	329	392	n/a	n/a	n/a
Difference	n/a	31%	8%	23%	n/a	n/a	n/a
Coarse Feed (Cruiser)	587.15	293.11	460.60	837.37	251.47	848.77	807.09
Coarse Feed (Sampled)	n/a	(342+361)/2=352	(314+285)/2=300	(215+207)/2=211	n/a	n/a	n/a
Difference	n/a	20%	54%	297%	n/a	n/a	n/a
Coarse Return (Cruiser)	587.15	293.11	460.60	837.37	251.47	848.77	807.09
Coarse Return (Sampled)	n/a	(342+361)/2=352	(314+285)/2=300	(215+207)/2=211	n/a	n/a	n/a
Difference	n/a	20%	54%	297%	n/a	n/a	n/a
Fine Feed (Cruiser)	331.54	325.9	321.74	424.88	312.67	427.17	368.81
Fine Feed (Sampled)	n/a	(188+209)/2=199	(241+211)/2=226	(252+228)/2=240	n/a	n/a	n/a
Difference	n/a	64%	42%	77%	n/a	n/a	n/a
Fine Return (Cruiser)	331.54	325.9	321.74	424.88	312.67	427.17	368.81
Fine Return (Sampled)	n/a	(188+209)/2=199	(241+211)/2=226	(252+228)/2=240	n/a	n/a	n/a
Difference	n/a	64%	42%	77%	n/a	n/a	n/a
Product (Cruiser)	389.56	351.45	365.86	491.05	347.23	491.83	566.58
Product (Sampled)	n/a	370.5	391	380	n/a	n/a	n/a
Difference	n/a	5.3%	6.9%	29%	n/a	n/a	n/a

3.4.5 Summary

CRUISER can predict the ACO gradation output within approximately 11% lower and 1% higher than the actual gradation. CRUISER predicts the ACR gradation output within 11% lower and 2% higher than the actual gradation. The Road Crush gradation output was predicted within approximately 16% lower to 0% higher than the actual gradation. Overall, CRUISER predictions of the output gradation are within ±10% error over 40 to 60% of the sieve sizes on any given sample. The error on the remaining sieve sizes can vary up to +2 and −16% on any given sieve size. Even with this range of accuracy, it is not sufficient to predict within product specifications used for the product types sampled. In reference to the tonnages predicted by CRUISER, it is evident that the program does quite well in predicting the final product tonnage within an average of 13.7% for only three samples.

3.5 Separate Component Analysis

As a means of analyzing the errors that propagate through CRUISER, when performing an analysis of an entire plant layout, a separate component analysis with respect to the actual data was completed. Basically, the actual input into the component was used as input into CRUISER for the component being analyzed and then CRUISER results were compared to the actual results. The four data sets used here are comprised of 3 ACO and 1 ACR product types.

It is important to note that CRUISER combines gradation streams in the following manner:

1. The cumulative percent passing of gradation #1 (i.e., Sample #1) is converted to TPH for each sieve size by multiplying the percent passing on each sieve by the TPH of that stream.
2. The cumulative percent passing of gradation #2 (i.e., Sample #2) is converted to TPH for each sieve size by multiplying the percent passing on each sieve by the TPH of that stream.
3. These two streams are then added and divided by the total TPH for the combined stream, which is just the sum of the TPH from gradation #1 and gradation #2. This gives the cumulative percent passing for the combined stream.

3.5.1 Coarse Crusher

A graph presenting the average results for two of the four possible samples is shown in Figure 3-5. Two samples had obvious erroneous gradation results and were omitted from the average calculation.

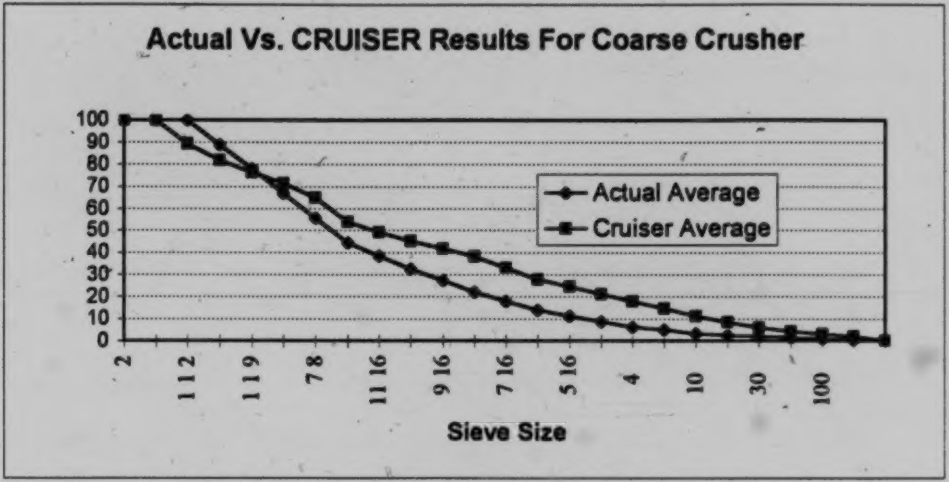


Figure 3-5 Actual vs CRUISER Results for the Coarse Crusher

Additional graphs for the coarse crusher are located in Appendix H. A few comments can be made for this crusher: The input gradations as obtained by the tester are fairly similar for all four samples However, two of the output gradations obtained by the tester are similar and two are not; the dissimilar two were omitted when an average of the results was calculated. The CRUISER output is very similar for all four samples. For the two more reliable samples, CRUISER predicts too coarse of a final product on the higher sieve sizes and too fine in the middle to lower end sieve sizes.

3.5.2 Fine Crusher

A graph presenting the average results for three of the four possible samples is shown in Figure 3-6. One sample was omitted due to questionable input gradation.

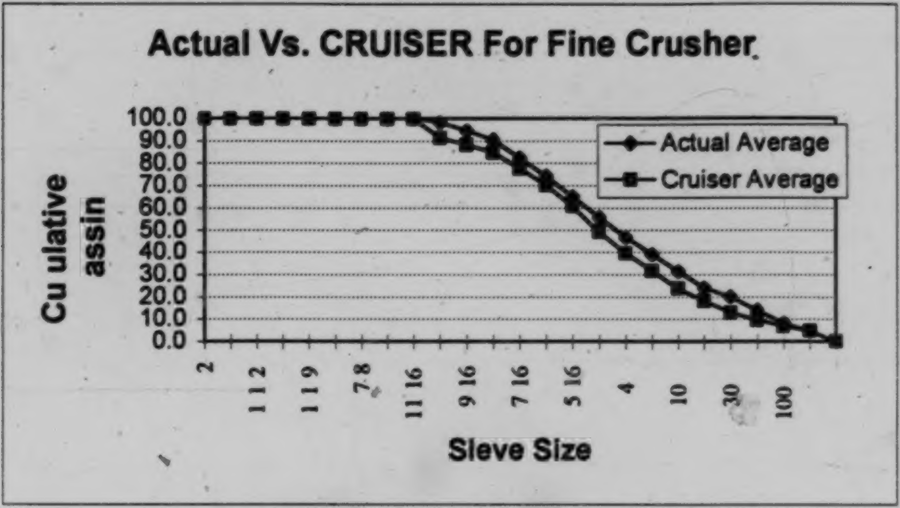


Figure 3-6 Actual vs CRUISER Results for the Fine Crusher

Additional graphs for the fine crusher are located in Appendix I. A few comments can be made for this crusher. The input gradations as obtained by the tester are fairly similar for three of the four samples. The CRUISER output for the four samples is very similar. For this crusher CRUISER predicts too coarse of a final product on all of the sieve sizes except for the last two sieve sizes. It is interesting to note that although CRUISER is more course that the actual product, it is only coarse by 7% passing at the most.

3.5.3 Scalper Screen

A graph presenting the average results for three of the four possible samples is shown in Figure 3-7. One sample was omitted due to questionable input gradation.

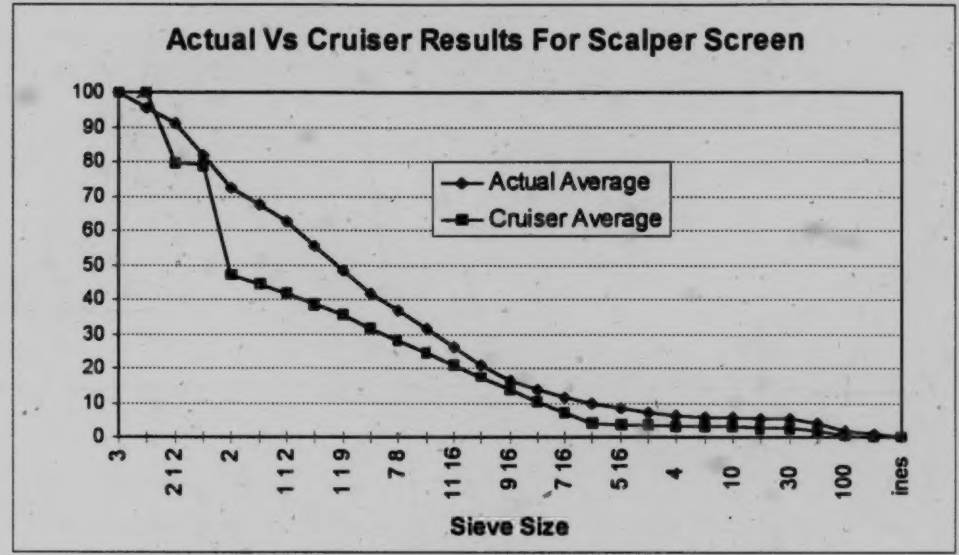


Figure 3-7 Actual vs CRUISER Results for the Scalper Screen

Additional graphs for the scalper screen are located in Appendix J. A few comments can be made for this screen. The input gradations as obtained by the tester are fairly different for all four samples. This is due to the high variability in the raw product as discussed earlier. A graph of the input gradation into the plant can be found in Appendix K. Three of the output gradations obtained by the tester are similar. One is not and was omitted when an average of the results was calculated. The CRUISER output for the four samples is fairly different on the higher sieve sizes, becoming increasingly similar as the gradations approach the smaller sieve sizes. For this screen, CRUISER predicts too coarse of a final product on all of the sieve sizes. Again, the comparison of CRUISER to actual samples might be distorted here because of the high variability in the raw product on the higher sieve sizes due to large rocks. It is interesting to note that CRUISER does predict the output quite accurately for one of the three decent sample results. On this sample, CRUISER predicts too fine by 11% passing and too coarse by 7% passing at the most.

3.5.4 Coarse Screen – Upper Deck

A few calculations had to be made to formulate the actual gradation of the input, which was also input into the CRUISER analysis. The plant layout can be referred to in Figure 3-1 for an easy understanding of how the streams were combined. The input into this screen was made up of the raw feed stream that had passed through the scalping screen, and the return feed stream from the coarse crusher. This was accomplished by the following process: The cumulative percent-passing on each sieve size was multiplied by the TPH for each respective stream. The two streams were then added together to form a combined stream. This combined stream was then divided by the total TPH from both streams, yielding a cumulative percent-passing gradation, which was then converted into a percent retained gradation. A graph presenting the average results for all four possible samples is shown in Figure 3-8.

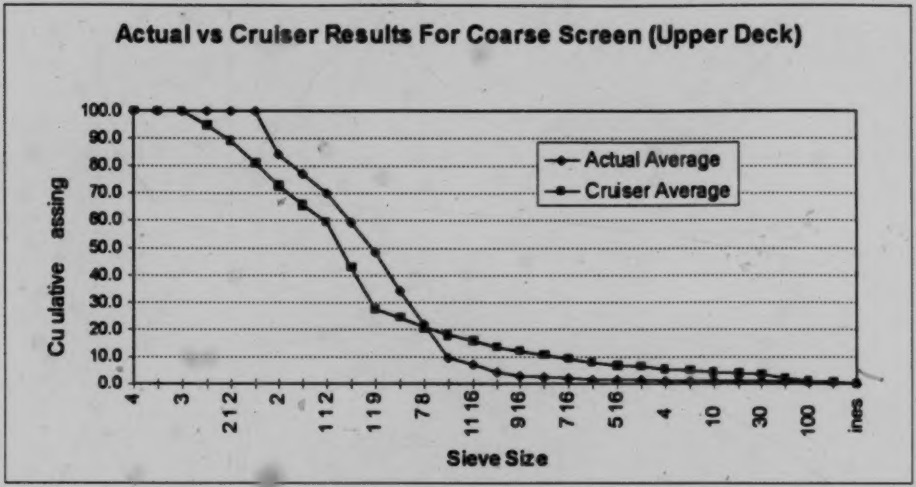


Figure 3-8 Actual vs CRUISER Results for the Coarse Screen - Upper Deck

Additional graphs for the coarse screen are located in Appendix L. A few comments can be made for the top deck of this screen. The input gradations as obtained by the tester are fairly similar for all four samples. These more consistent results are likely due to the addition of the crushed product from the coarse crusher combining with the higher variable raw product at this sampling location. All four of the output gradations obtained by the tester are similar and were included when an average of the results was calculated. The CRUISER output for the four samples is a little different on the higher sieve sizes and becomes increasingly similar as the gradations approach the smaller sieve sizes. CRUISER predicts too coarse of a final product on the larger sieve sizes and a little too fine on the smaller sieve sizes. Only with the second sample did CRUISER predict too fine of a product for a few sieve sizes. On this sample, the actual gradation as obtained by the tester varies from the actual results of the other three samples. The comparison of CRUISER to actual samples may be slightly distorted because of the high variability in the raw product on the higher sieve sizes due to large rocks being present in the samples. The return product being fed into this screen by the coarse crusher slightly reduces the variability. Given the samples obtained from the top deck of this screen, it is evident that CRUISER predicts too coarse of a product on higher sieve sizes and too fine of a product on lower sieve sizes.

3.5.5 Coarse Screen – Lower Deck

Due to the dependence of analysis of this deck on output streams that could not be sampled, accurate graphs revealing the efficiency or gradation were not created. However, the level of actual efficiency of this deck's performance could be approximated. Samples from an input stream into the second crusher included the combined streams of the oversize from the lower deck on Screen #2 and the oversize from both decks on Screen #3. So the evaluated efficiency of this sample is actually a combined efficiency of three decks from two screens. Table 3-7 shows the efficiencies for the four samples using the limiting sieve size of 0.4375 inches.

Table 3-7 Coarse Screen – Lower Deck Efficiency

	Sample #1	Sample #2	Sample #3	Sample #4
% Passing > 0.4375 inches	84	83	54	84

It is evident that for three out of the four samples, the efficiency of these three decks combined is approximately 83 to 84 %. One thing to note is that, from observations when collecting samples, the two decks on the third screen were deemed to be operating quite efficiently. The restricting sieve size on the second deck of this screen was 0.5625 inches. So the majority of the inefficiency from the screening process of this combined stream is due to the lower deck on Screen #2. It is not possible to determine exactly what the inefficiency is, although it is likely to be in the 12 to 14 % range out of the possible 16 % inefficiency known to exist.

3.5.6 Fine Screen - Upper Deck

It has been determined that there are too many unknowns surrounding this screen to perform a reliable analysis. Additional sample locations for the obtained samples are required to determine unknown gradations and/or stream capacities in TPH. Analysis of this screen was attempted by checking how much material was larger than the screen size of the lower deck of Screen #3 in the input stream into Crusher #2. This could possibly evaluate the amount of undersize (i.e., material passing through the screen) in the oversize (i.e., material remaining on top of the screen) from Screen #3. Analysis of the data did not result in an accurate efficiency for Screen #3. The increase in percent passing the 0.563 inch lower deck screen size of the four samples is presented in Table 3-8 below.

Table 3-8 Fine Screen – Upper Deck Efficiency

% < 0.563 inch	Sample #1	Sample #2	Sample #3	Sample #4
% Passing Input	60	60	33	54
% Passing Output	88	88	88	88
% Difference	28	28	56	33

It is evident from the data that no strong conclusions could be made about the efficiency of Screen #3. However, we can conclude that the average percent passing increase on this sieve size is about 36% using all four samples and 30% using three of the samples.

The analysis of this screen beyond what is shown here is not attainable without additional gradation samples. Only one more sample location of a possible two would be required, however, sampling from either of these two streams was physically impossible to obtain in the same manner as the others.

3.5.7 Fine Screen – Lower Deck

Accurate graphs revealing the efficiency or gradation for the lower deck could not be created due to the lack of sample locations required to perform the analysis.

3.5.8 Results

It is difficult to generalize the results based on just crushing or screening characteristics. Not every crusher or screen produced results which were consistent from like component to like component (i.e., from Crusher #1 to Crusher #2). In terms of the crusher analysis, it can be said that CRUISER generally predicts too coarse of a product gradation on the higher sieve sizes regardless of whether a coarse or fine crusher is used. However, with the middle to lower sieve sizes, CRUISER predicts too fine of a product with a coarse crusher and too coarse of a product with a fine crusher. The discrepancy in error between actual and CRUISER predicted results is greater with the coarse crusher in the middle to lower sieve sizes. In terms of the screening analysis, it can be said that CRUISER has difficulties with predicting product gradations accurately when there is high variability in the input stream. This can also be due to the sampling process, where it is not possible to sample the input and output stream from the same relative sample, thus the accuracy of the actual results will be affected. This apparent inaccuracy can be seen quite easily on Screen #1 and on the upper deck of Screen #2. A source of error for the screening process is the efficiency determination portion of the program. It has become evident that CRUISER predicts a lower screening efficiency than what is actually achieved.

3.6 Conclusion

The data that was collected from an actual aggregate plant was assessed and used to determine as much as possible the accuracy and inefficiencies of CRUISER. One major difficulty in obtaining data was due to the difficulty of retrieving aggregate samples safely from the conveyor belts of screens. Some of these conveyor belts were located under the screen itself and accurate sampling was impossible. The second major difficulty in obtaining meaningful data was that the aggregate plant had to be shut down in mid-stream to allow for sampling of several product streams. Starting up when loaded with aggregate is not something the aggregate plant is designed for. After the first case of equipment failure when starting up the plant after obtaining samples in mid-stream occurred, the sampling of intermediate aggregate streams was no longer allowed.

Gradation data from all streams is the most meaningful data to analyze CRUISER. It allows for specific detection of gradation errors within individual processes in the program and the causes for them. Overall, CRUISER is fairly accurate when predicting the final product gradation with some error. The final product gradation is within ±10% error over 40 to 60% of the total number of sieve sizes on any given sample. The final product gradation predicts up to 16% coarser and 2% finer than the actual results over all samples. Even with this range of accuracy, it is not sufficient to predict within product specifications used for the product types sampled. In reference to the tonnages predicted by CRUISER, it is evident that the program does quite well in predicting the final product tonnage within an average of 13.7%. Analyzing individual equipment components of the plant presented a more detailed analysis of where inaccuracies in simulation analysis could be found. The result of the coarse crusher analysis is that CRUISER generally predicts too fine a gradation. The result from analyzing the fine crusher is that the gradation is generally predicted too fine by the CRUISER program but with less error than the coarse crusher. When looking at the screening results, it became evident that the scalper screen results were quite variable. Analysis was difficult because the raw feed was quite variable, being mostly sand with some large rocks. The sampling of before and after this screen is likely the main cause of most of the error. The results from the next screen are more indicative of where screening errors might be occurring. It is evident that on the top deck that CRUISER predicts too coarse on the larger sieve sizes and too fine on the smaller sieve sizes.

4.0 Simulation Analysis of CRUISER

4.1 Introduction

Improvements to the CRUISER program could occur in the crushing and screening processes. Each of these options were explored and evaluated to attain additional accuracy for each of these processes. For the crushing process, only two parameters could be modified to change the gradation. One was where the cutoff of which sieve sizes were absolutely crushed; the other of which remained uncrushed. For the screening process, an area of improvement is the screen efficiency calculation. Evaluation of this area and improvements to the program will be implemented in this chapter.

4.2 Crusher Sensitivity

The effectiveness of the crushing process is measured by the ability of the crusher to break down the aggregate to a specified smaller maximum aggregate size. The ability of a crusher to achieve this may be affected by several operating conditions, such as high moisture content, winter conditions and crusher overloading. The following algorithm models the crushing process within CRUISER:

1. If there is no oversize (i.e., material larger than the crusher setting) in the input stream to the crusher, then the output stream is the same as the input stream.
2. If the input stream load exceeds the crusher capacity, an error is reported after analysis is completed.

3. Calculate the effective tonnage to crush. The effective tonnage is the sum of the materials larger than half the crusher setting and smaller than one and a half the crusher setting.
4. Set the initial output gradation to the expected gradation as given by empirical tables.
5. If the amount of oversize in the output exceeds that of the input, then adjust the oversize in the output.
6. Add material that is unaffected by the crushing process (i.e., material less than 0.5 x the crusher setting) to the output gradation.

For the crushing operation within CRUISER, the sensitivity analysis will be approached in the following manner: First, three samples will be tested over all of the crusher settings available within CRUISER. The main assumptions with respect to the crushing process within CRUISER at the present time are the following:

1. All material above 1.5 x the crusher setting will be crushed.
2. All material below 0.5 x the crusher setting will remain uncrushed.

For this reason, one of the samples will be considered a coarse sample and weighted on the upper end to test the first assumption. This sample will be referred to as a high weighted sample. The second sample will be in between the two assumptions, consisting of a more nominal or typical gradation. This sample will be referred to as a medium weighted sample. The third sample will be considered a finer sample and weighted on the lower end to test the second assumption. This sample will be referred to as a low weighted sample.

Each of these samples will be put through a chosen setting a number of times while varying the 1.5 and 0.5 factors. The 1.5 factor will be varied from 1.0, 1.25, and 1.5. The 0.5 factor will be varied from 0.25, 0.5, and 0.75. While one factor is being varied, the other will remain fixed (either 0.5 or 1.5).

Since there are 10 crusher settings and 5 combinations of factors for each crusher setting, the result is a total of 50 output gradations for each sample gradation. This comes to a total of 150 gradations from all three samples. For ease of evaluation, 3 sets of crusher settings were compared, where the lowest setting is 0.375, the middle setting is 1.0, and the highest setting is 2.0.

4.3 Crusher Analysis

4.3.1 Crusher Setting

A typical example of a graph showing the resulting gradations after crushing with different crushing factors is shown in Figure 4-1. Additional graphical results can be found in Appendix M. One general observation is that the range of the output gradation curves increase as the crusher setting increases.

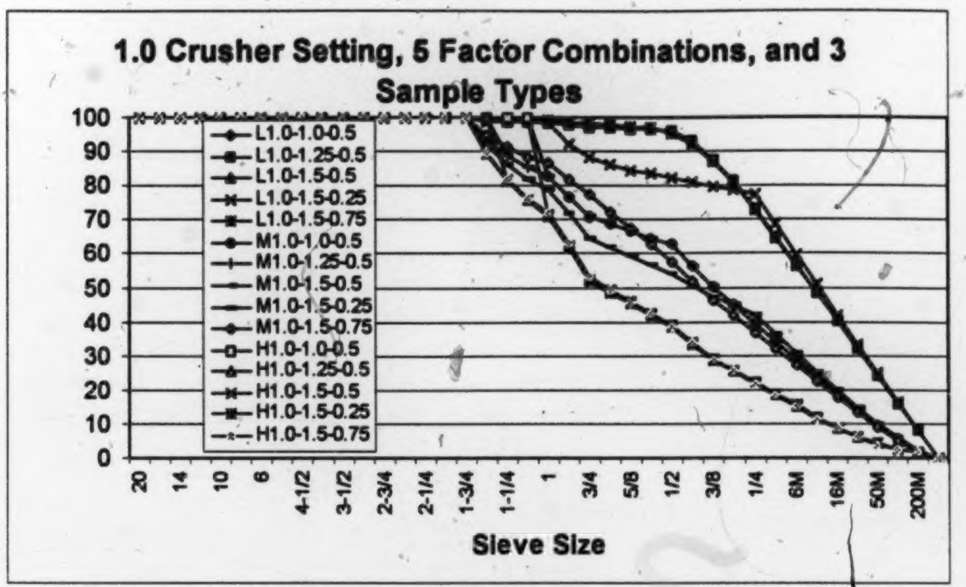


Figure 4-1 Various Crushing Parameters on Three Typical Gradations

With all three crusher settings (i.e. 0.375, 1.0, and 2.0), adjusting the 0.5 factor up or down does very little to change the gradation in most cases, regardless of whether the gradation is low, medium or highly weighted. With all three crusher settings, varying the 1.5 factor does indeed affect the upper end of the gradation curve. This factor controls the last sieve size in which 100% passing occurs.

A high weighted sample will not vary in the upper end by a low crusher setting (however, this is not a realistic case since a crusher at this setting would not be used on such a finely graded material). The opposite is true, where a low weighted sample will not vary in the lower end by a high crusher setting.

4.3.2 Crushing Factors

Table 4-1 shows the variation in percent passing when using factor combinations other than 0.5 and 1.5.

Table 4-1 Percent Passing Error of Lower and Upper Bound Factors

Sample	0.375	0.375	1.0	1.0	2.0	2.0
	Max. (-ve)	Max. (+ve)	Max. (-ve)	Max. (+ve)	Max. (-ve)	Max. (+ve)
Low-1.0-0.5	0	5	0	2	0	1
Low-1.25-0.5	-3	3	0	0	0	0
Low-1.5-0.25	-9	4	-14	4	-2	0
Low1.5-0.75	-4	2	-1	1	0	0
Med-1.0-0.5	0	11	0	15	-5	10
Med-1.25-0.5	0	6	0	6	-6	6
Med-1.5-0.25	-4	2	-8	3	-14	3
Med1.5-0.75	-2	2	-5	6	-3	2
Hgh-1.0-0.5	0	14	0	24	0	25
Hgh-1.25-0.5	0	8	0	10	0	14
Hgh-1.5-0.25	0	0	0	0	-1	0
Hgh1.5-0.75	0	0	0	0	-4	6

From Table 4-1 we can see that the major variance from using the factors of 1.5 and 0.5 occur when the upper bound factor is 1.0, or when the lower bound factor is 0.25. The value of 1.0 for the upper bound factor is impractical since it is not possible that 100% of the material will be crushed to the crusher setting or below it, unless no crushing occurs, which is also impractical. The value of 0.25 for the lower bound factor could be impractical as well. This would mean that 25% of the material of less size than the crusher setting will not be further crushed. It would also mean that the remaining 75% would be reduced in size due to inter-particle attrition.

4.3.3 Lower Bound Factor – 0.5

Significant deviations only occur in the middle portion of the gradation when the 0.5 factor is changed to the extremes of either 0.25 or 0.75. The 0.25 factor results in a significantly lower gradation in three out of the nine possible samples with the 0.5 factor set at 0.25. This is because CRUISER crushes everything above 0.25 x the crusher setting and references the exact input for every sieve size below this setting. Only once out of nine times does the 0.75 factor cause the gradation to become higher. This is where the medium gradation is used in combination with a nominal crusher setting of 1.0, a typical crushing situation. The variance increases up to +6% passing in the upper middle of the gradation and down to –5% passing in the lower middle of the gradation. This might be an important result and therefore might warrant a change in the lower bound factor from 0.5 to 0.75 if the accuracy of CRUISER results are improved to better match actual results.

Based on an evaluation of the limited number of samples obtained from an actual plant, the value for the lower bound factor should be changed from 0.5 to 0.75. This change, however, will result in an accuracy gain of less than 1% on any given sieve size upon which it has an effect. This factor would have a greater influence on finer graded materials; however, this type of material is not typically further crushed. Analysis was done using ACO sample #1 and ACR sample #4 on both the coarse and fine crushers. Since it was found that only a modest gain in accuracy would be achieved in pursuing this factor, the evaluation was stopped after two samples.

4.3.4 Upper Bound Factor – 1.5

Varying the 1.5 factor to 1.0 or 1.25 does indeed affect the upper end of the gradation curve. This factor controls the last sieve size in which 100% passing occurs. Setting the factor to 1.0 is not realistic, since it is not possible for all oversize material to be crushed by the cone crusher. This factor was adjusted to 1.0 was used for sensitivity purposes only.

Table 4-2 details the range in which the upper bound factor should fall in order to match the actual sieve results for all four samples when evaluated with both fine and coarse crushers. This particular factor will only determine where the last sieve with 100% passing will be. The next sieve down will be read from the table of gradations for the crusher setting specified. All remaining sieve sizes will not be affected by this factor.

Table 4-2 Range of the Upper Bound Factor

	ACO Sample#1	ACO Sample #2	ACO Sample #3	ACR Sample #4
Coarse Crusher	1.3 to 1.45	1.3 to 1.45	1.3 to 1.45	1.8 to 1.95
Fine Crusher	1.25 to 1.28	1.29 to 1.55	1.45 to 1.55	1.45 to 1.55

Based on an evaluation with the limited number of samples obtained from the actual plant, the recommended value for the upper bound factor of 1.5 is 1.45. This value will satisfy 6 out of the 8 samples so that they properly match the last sieve size to have 100% passing. One correlation that could be seen here is that the higher the crusher setting, the lower the range of what the 1.5 factors should be. Conversely, the lower the crusher setting, the higher the range of factors results. With these eight samples, the value of

1.45 fits into the corresponding range for six samples. This value may only be representative of this particular plant or aggregate pit. Changing either factor to the values found here may not necessarily improve the gradation results of the program for all future cases.

4.3.5 CRUISER Program Improvements

As a meaningful addition to CRUISER which will improve its accuracy, the researcher suggests that the program interface be modified with respect to these factors. This will allow the user to input different upper and lower bound factors to better reflect the aggregate production model in question, or the aggregate pit from which the product is being produced. To do this, the user must calibrate the model before using it to find out what combination best represents the actual data for the specific plant arrangement. While this improvement does make the program more accurate when compared to actual results, the increase in accuracy does not warrant trading off the simplicity of the crushing model. Making the user interface portion of the program more complicated will only deter the industry from using the program. As a result, the program improvement suggested in this section should be postponed until it is deemed necessary for slight accuracy improvements to be made to the crusher modeling within CRUISER. Greater additional accuracy for this process will be achieved through the use of neural networks; these will be discussed in Chapter 5.

4.4 Screening Efficiency

Screening efficiency is a measure of the effectiveness of the ability of a screen to separate an input stream into its respective undersize and oversize aggregate streams. Efficiency of a screen can be affected by several operating conditions; such as moisture content, screen angle, and screen overloading. A 90% efficiency value indicates that 10% of the undersize material failed to pass through the screen openings and remained in the oversize stream. Typical values for screen efficiency can range from 60 to 100%.

The aggregate plant observed had three screens, the first of which processed the raw feed into the plant. The primary function of this first screen is to remove the majority of the fines (i.e., sand) from the raw product. Determining the efficiency of this screen is not generally reliable since it directly depends on the raw feed, which varies widely. However, analysis will be performed on this screen because the results may still be meaningful. A portion of the second screens will be analyzed for more reliable efficiency results. For reasons explained later, efficiency analysis was not performed on the second portion of the second screen as well as the third screen. CRUISER determines each screen's efficiency individually. At the observed aggregate plant, efficiency was evaluated by determining the amount of undersize (i.e., aggregate smaller than a given screen size) in the oversize (i.e., aggregate larger than a given screen size). To calculate actual screen efficiency, the following procedure was executed:

1. Determine the amount of undersize in the oversize stream by weight in grams.

2. Add this weight to the undersize weight to get the total undersize weight. (Note: When combining streams, match the total weight for both streams so that they are equal before adding their weights together).
3. Divide the amount of undersize in the oversize by the total undersize.
4. Multiply by 100% to determine the screen inefficiency or subtract this amount from 100% to get screen efficiency.

Within CRUISER, screening efficiency is directly dependent on a loading ratio. This ratio is a relationship between the actual load on the screen and the rated capacity of the screen. This relationship as shown in Figure 4-2 was developed by Allis-Chalmers (Hancher and Havers, 1972) and is the current basis of screen efficiency analysis within CRUISER.

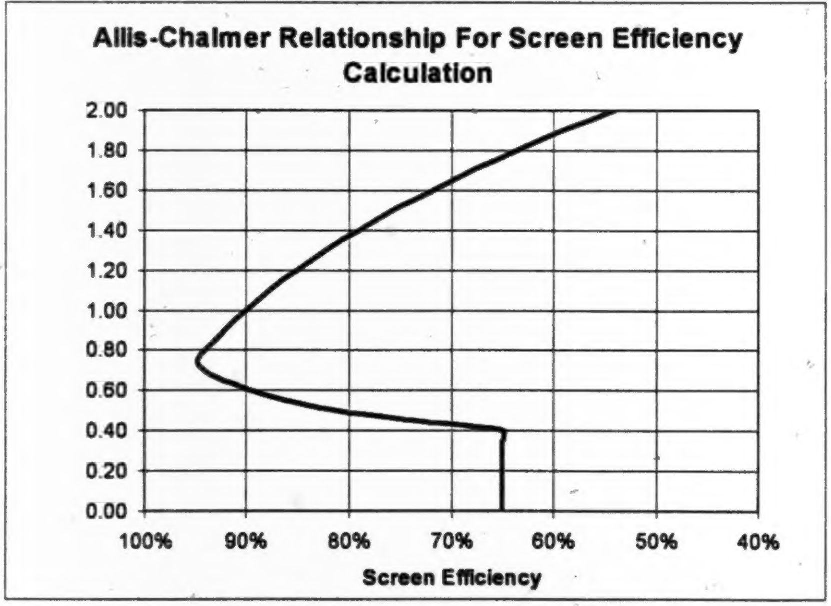


Figure 4-2 Allis-Chalmers Screen Efficiency Curve

The rated capacity of the screen is based on a formula that takes a basic screen capacity per square foot and multiplies it by the surface area of the screen deck being evaluated. This number is then multiplied by a number of factors: screen incline, deck number, gradation oversize and undersize percentages, material condition, material particle shape, percent open area of the screen, and screen hole shape. The values for these factors are obtained from an aggregate production handbook (Cedarapids, 1984). Since the actual load also affects how CRUISER determines the efficiency, actual recorded TPH amounts will be entered for each specific sample. Once screen efficiency is calculated, the size distribution of undersize particles remaining in the oversize stream is determined using developed relationships (Hanchers and Havers, 1972). Previous experience shows that the undersize material is equally distributed above the optimum loading ratio. Below the optimum loading ratio, the percentage of undersize material in the larger sizes increases exponentially. A graph of the relationship used to determine the distribution of the undersize in the oversize when the loading ratio is less than 0.75 is shown in Figure 4-3.

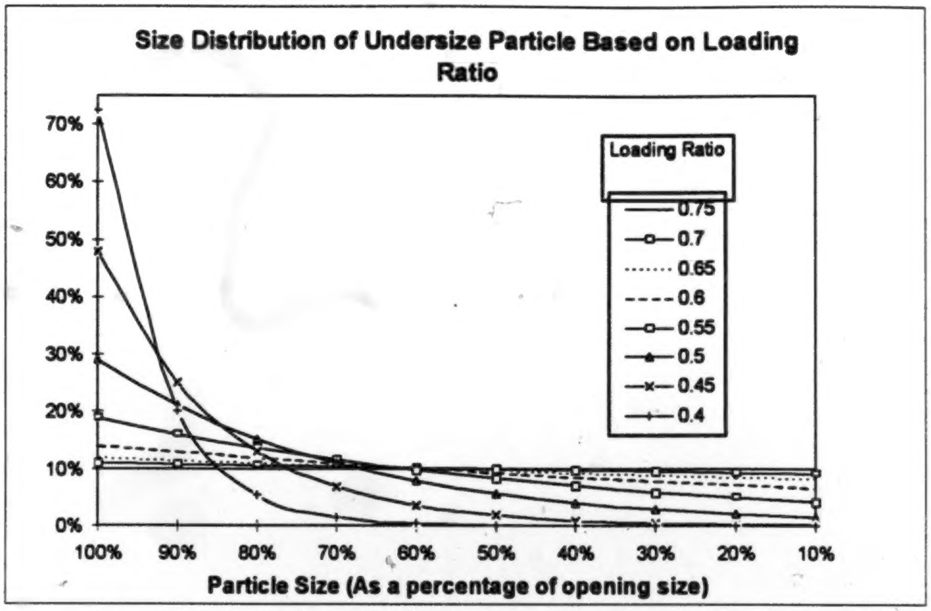


Figure 4-3 Distribution of Undersize In Oversize Based On Loading Ratio

The algorithm for the screening process is as follows:

1. Calculate rated capacity for the screen.
2. Calculate loading ratio using the rated capacity and the input stream load.
3. Calculate efficiency using the Allis-Chalmers relationship.
4. Calculate undersize and oversize streams based on 100% efficiency.
5. Adjust for inefficiency by removing appropriate amounts, for each size, from the undersize stream and adding them to the oversize stream. If the loading ratio is greater than 0.75, then the size distribution of the undersize material is equally distributed. If the loading ratio is less than 0.75, then the distribution of the undersize material is determined using Figure 4-3.

For the first sample (ACO #1) the method of determining the TPH for the measured aggregate streams was not yet developed. For this sample, the average of the two other samples of this product type will be used as an approximation. Table 4-3 outlines the input tonnage per hour for each of the three screens and four samples.

Table 4-3 Screen Input TPH

	ACO #1	ACO #2	ACO #3	ACR #4
Screen #1	533	538	528	598
Screen #2 (screened pitrun + coarse return)	617	(260+360) = 620	(329 +285) = 614	(391 + 207) = 598
Screen #3	210	209	211	228

The greatest difference between the minimum and maximum TPH values for a given screen are: 13% for Screen #1, 4 % for Screen #2, and 9% for Screen #3. The greatest difference is in the first screen, which is expected since the TPH determination was based on the weight of a relatively small sample. Also, this sample being of the raw feed to the plant is highly variable. The evaluation of this difference between TPH calculations determined from the plant was important since these TPH values were necessary to compare actual and CRUISER predicted screen efficiencies.

4.5 Screening Analysis

When evaluating CRUISER on an individual component basis, one input is the actual load in tonnes per hour. Varying this can have a direct result on the efficiency calculated using the Allis-Chalmers relationship, and therefore the gradation results as well. A trial test was done using the ACO Sample #1 input gradation into Screen #2 and varying the feed rate. Both the efficiency used by CRUISER for the top screen deck and the output gradation were recorded to evaluate whether or not a particular efficiency affects the accuracy in a positive way. The accuracy was measured in terms of absolute total error on all sieve sizes. Table 4-4 contains the results of this experiment.

Table 4-4 Efficiency Sensitivity

Actual Load (TPH)	Rated Capacity (TPH)	Ratio = AL/RC	Allis-Chalmers Efficiency (%)	Absolute Total Error (% Passing)
600	322	1.86	61	160
500	322	1.55	73	138
400	322	1.24	84	122
300	322	0.93	91	117
200	322	0.62	91	125

The gradations from each of the test runs were compared with the actual results. It was found that the best results were obtained when the efficiency was the highest at 91%. This reveals that the output gradation is directly dependent on the efficiency of the screen, as determined by CRUISER during analysis. One important point to note is that the efficiency graph CRUISER uses is based on a rated capacity, which is somewhat based on the input gradation as illustrated below.

An analysis was performed using CRUISER and screen #2 only. The efficiency of both the bottom deck and (most importantly) the top deck was evaluated. The output from the analysis is presented in Table 4-5.

Table 4-5 Efficiencies Calculated by CRUISER

Item	Description	Sample #1	Sample #2	Sample #3	Sample #4
Top Deck	Actual Load	620	610	630	600
	Rated Capacity	326	340	515	495
	% Loaded	190	179	122	121
	% Efficiency	59	64	84	85
	Actual Efficiency	98.8	98.8	98	98.6
	% Difference	40	33	14	13.6
Bottom Deck	Actual Load	244	255	364	318
	Rated Capacity	303	296	309	413
	% Loaded	80	86	118	77
	% Efficiency	94	93	86	94
TPH	Top Deck	376	355	266	282
	Bottom Deck (oversize)	189	199	230	173
	Bottom Deck (undersize)	54	56	134	146

From the above table one can see that the efficiency used by CRUISER is quite variable for the top deck (59% to 85%) and fairly consistent for the bottom deck (86% to 94%). Comparing the actual efficiency of the top deck to the efficiency predicted by CRUISER reveals that there is a discrepancy of about 14% for two of the samples and 33 to 40% for the other two samples. One important point to note is that the only variables that change from sample to sample are the input load in TPH and the input gradation. It is interesting to note that Samples #1, #2, and #3 are all within 6 TPH of each other, yet Sample #3 has a remarkably different efficiency than the other two. This is a result of the variance in the rated capacity for each sample, which is based on the calculation of a basic capacity multiplied by a number of factors. Some of these factors are based on the gradation of the input material, which results in a wide variance in different rated capacities. This can be corrected by both using the existing efficiency curve and determining new rated capacities, or by using the existing rated capacities and developing a new efficiency curve. Either way, numerous data points will be required to determine an adequate range of rated capacities or develop an efficiency curve. This data is not available at the moment but will be required to find a solution to one of these two options.

In reference to the actual efficiency of the screen, 98.8% efficiency means that 1.2% of the undersize material was found to be in the oversize stream from this screen. It can be said that CRUISER underestimates the efficiency of the screening process, thus affecting the gradation results in a negative manner. The researcher suggests that the efficiency determination within CRUISER be changed to better reflect the actual results of efficiency and gradation. This is to be done by allowing the user of the CRUISER program to define screen efficiencies for each screen deck, thus improving the accuracy of the screening analysis.

4.5.1 CRUISER Program Improvements

As previously mentioned, either the existing efficiency curve or the rated capacity determination must be modified. Either way, numerous data points will be required to either determine an adequate range of rated capacities or develop an efficiency curve. This data is not easily obtainable and is necessary for finding a solution to one of these two problems. The most viable solution would be to leave the model the way it is and give user the option to override the suggested efficiency of the screen. This efficiency could be a factor based on past experience or historical information.

The last option was chosen, allowing the user to override the calculated efficiency for any deck of a screen. If the user gives no efficiency input, then the default is to determine the efficiency by calculated means as done previously using the Allis-Chalmers relationship. A simple experiment was done to evaluate the accuracy improvements. This was done by matching actual results with four samples, using the screen efficiency with minimum error for Screen #2 deck one. The total absolute error and average absolute error for all four samples is given in Table 4-6. An important note is that the average absolute error is taken from the total absolute error and divided by the number of gradation sieve sizes containing material, which was, on average, 26 for the four samples.

Table 4-6 Gradation Error Using Actual Screen Efficiency

	ACO #1		ACO #2		ACO #3		ACR #4	
	Allis-Chalm Eff.	User Defined Eff.	Allis-Chalm Eff.	User Defined Eff.	Allis-Chalm Eff.	User Defined Eff.	Allis-Chalm Eff.	User Defined Eff.
Efficiency	59	80	64	85	86	86	94	95
Total Absolute Error (%)	165	136	194	160	230	230	294	290
Average Absolute Error (% per sieve)	6.3	5.2	7.5	6.15	8.8	8.8	11.3	11.1

It is clear that for three of the four samples tested, an accuracy improvement does occur. However, this accuracy improvement has a minimal effect, decreasing the error by an average of 1 % for each sieve for two of the four samples. Although this seems good, it is not enough improvement to bring the CRUISER predicted results within gradation specifications. The graphs comparing the actual results for various user input efficiencies can be found in Appendix N. It is evident that certain chosen efficiencies model the actual gradation better than others, although none are close enough for any sieve size.

Another comparison was done with utilizing the option of manually changing the screen efficiencies. This time, the screen efficiency was changed until the tonnage per hour reflected what was obtained on site for the output stream of the top deck of Screen #2. The results of the comparison can be found in the Table 4-7.

Table 4-7 Calculated Efficiency with Accurate TPH

	Sample #1	Sample #2	Sample #3	Sample #4
Measured TPH	330(estimated)	350	305	205
CRUISER Matched Efficiency (%)	75	65	75	N/A

These new efficiencies varied from the optimum efficiencies differed from 5 to 20% in terms of average absolute error. However, this is a slight growth in error from CRUISER predicted efficiencies, which varied by 0 to 15% across all four samples. As mentioned before, CRUISER predicts the final output tonnage within 14% but is not as effective in predicting intermediate streams, which were used in this comparison.

4.6 Conclusion

With respect to improvements to the CRUISER program, changes were made to the crushing process to enhance gradation accuracy. An attempt was made to improve the accuracy by adjusting two cut-off factors; some gain in accuracy occurred. The research in the next chapter will add more accuracy to this process within CRUISER than what the adjustment of these two cut-off factors could. The cut-off factors could vary for each plant or aggregate pit and would likely require calibration. A neural network could incorporate these changes much more accurately and model more factors than these two cut-off factors. The screening process had a weakness in the way the efficiency of a screen was calculated. This modification to the screening efficiency will have an affect on gradation and TPH calculations. The user can select from a calculated efficiency or input an efficiency value for each screen deck of each screen in the CRUISER program. The user defined efficiency can be obtained from a plant operator who is familiar with the plant equipment and product being produced or it can be achieved by obtaining samples from a screen and determining the actual efficiency. For these reasons a more area-specific and plant-specific approach should be taken. The following section will discuss how this can be done using a form of artificial intelligence called neural networks.

5.0 Developing A Crushing Neural Network For CRUISER

5.1 Introduction

In this chapter a neural network is presented. This model will replace the algorithmic model used in CRUISER. A neural network accepts more variable input and creates output data that more accurately reflects actual results. Presently, the "CRUSH" routine accepts variable data as input but the output is mainly dependent on the crusher setting. As demonstrated in chapter 4, the number of sieve sizes has a slight bearing on the output gradation. The crusher setting basically determines which set of empirical data will be chosen as a basis for the output gradation. The number of variables in the neural network reflects the amount of individual data coming into and going out of the existing "CRUSH" routine (i.e., 40 sieve sizes for input and output). Throughout the development of a crushing neural network, the eventual incorporation into CRUISER was kept in mind so this research would be both meaningful and useful. This chapter will include a prototype model, several intermediate models, and a final neural network model. Guidelines on how to create more of the same data for the training of additional crusher types or crusher settings are included as well.

5.2 Crushing Neural Network - Prototype

5.2.1 Sample Development

The development of a neural network requires sufficient amounts of data to represent the problem properly. The data needs to include the global spectrum of what the network is expected to predict. The network will not be able to predict the crushing of coarse samples if we only train it with finer samples. It would be ideal data to develop a crushing neural network with actual crushing data. Since large amounts of data are required to develop a meaningful model, simulation was used to create variation in the data. Simulation was used because data could not be obtained from local aggregate producers or aggregate production manufacturers.

Training a neural network requires sets of input and output. To obtain several sets of input and CRUISER processed output, a Visual Basic program was developed. The program essentially generates sample gradations, converts these numbers into a form for the CRUISER "CRUSH" routine, process the data, and stores both the gradation input and output in an Access database. The data is then copied into Excel and converted into a file that can be read by Neural Works Explorer. A neural network will then be trained and tested to evaluate its accuracy. The code of the program can be found in Appendix O.

A total of 500 sample gradations were obtained using this program. Each of these 500 samples were processed by the "CRUSH" routine from CRUISER 10 times, once for each of the 10 possible cone crusher settings available in CRUISER. The overall sample total was 5000 for both input and output data. A plot of 250 gradation samples can be found in Appendix P. The primary model is created for a cone crusher, but could be easily modified to create data for other types of crushers within CRUISER. Manufacturers of aggregate production equipment recognize that cone crushers are affected by more factors than other crusher types (Pioneer, 1996). Cone crushers are more commonly used in practice, so a trained neural network representing this crusher type would be most beneficial. The following reasons represent the large variability in empirical data required to adequately model cone crushers:

1. Capacities and product gradations produced by cone crushers will be affected by the method of feeding, characteristics of the material, speed of the machine, power applied, and other factors.
2. Properly controlled, continuous, uniform feeding of material around the feed opening of a cone crusher is essential for maximum production.
3. Hardness, compressive strength, mineral content, grain structure, plasticity, size and shape of feed particles, moisture content, and other characteristics of the material affect production capacities and gradations.
4. The minimum closed side setting (CSS) of the crusher may vary. This is the smallest size opening possible for material to pass through as the mantle of the crusher rotates against the bowl. The CSS may be greater than listed since it is not a fixed dimension. It will vary depending on crushing conditions, the compressive strength of the material being crushed, and stage of reduction. The actual CSS is the setting just before the bowl assembly lifts minutely against the factory recommended pressurized hydraulic relief system.
5. The manufacturer's data is based on a 20% recirculating load in a closed-circuit crushing cycle. The manufacturer also specifies that the screen opening for a closing cycle must be something larger than crusher CSS to control recirculating to a maximum of 20%. The data collected in this research indicated that the circulating loads were in the range of 24 to 37%. Due to the variability of rock sizes in the raw feed stream, it would be impossible to control the recirculating loads near a certain percentage to be able to use existing manufacturing crushing data.

These factors could also be included within the neural network model when actual data can be obtained.

5.2.2 Neural Network Architecture

The neural network used was a feed-forward, back-propagation 3-layered neural network. The number of variables the "CRUSH" routine within CRUISER presently uses governs the number of input and output variables. This was done so the developed model could replace the "CRUSH" routine. The neural network architecture consists of 41 input nodes, 20 hidden layer nodes, and 40 output nodes. The input nodes are made up of a crusher setting and the weight retained on each of 40 sieve sizes. The output nodes are made up of the weight retained on each of the 40 sieve sizes after being processed by the "cone crusher".

5.2.3 Optimum Training Parameters

Due to the length of time it took to train one network, only one combination was executed using Neural Works Explorer. The Delta learning rule, the Sigmoid transfer function, and a learning rate of 0.8 was used to train the model. The neural network was trained with a total of 750,000 iterations when it was certain that the error of the network had stabilized and the error on any given node was less than 0.0005.

5.2.4 Neural Network Weights

The network weights can be found in Appendix R. These weights could be hard coded into a program if so desired to produce the same accuracy and results which have been found in this research.

5.2.5 Neural Network Accuracy

The network was trained with 4750 sample gradations and tested with 250 sample gradations. The testing set amounted to 5% of the total generated sample set. The average absolute error for each of the 250 test samples is shown in Figure 5-1. The range of the absolute error is from 2 to 7 grams or 3 to 11%. The minimum error of any given sieve was 0 grams; the maximum error was 82 grams. The average of the absolute error across all 250 samples was 4.5 grams per sieve size. This translates into 4.5 grams x 40 sieve sizes for a total of 180 grams of error for any given sample. The average sample size is approximately 2500 grams, which translates into this network as having an average error of 7.2%. This is an acceptable amount of error considering the randomness of the samples created and used to test the network. One important thing to note is that the neural network was trained using output data from the "CRUSH" routine. This shows that the neural network can replace the existing crushing module within CRUISER with allowance for some error.

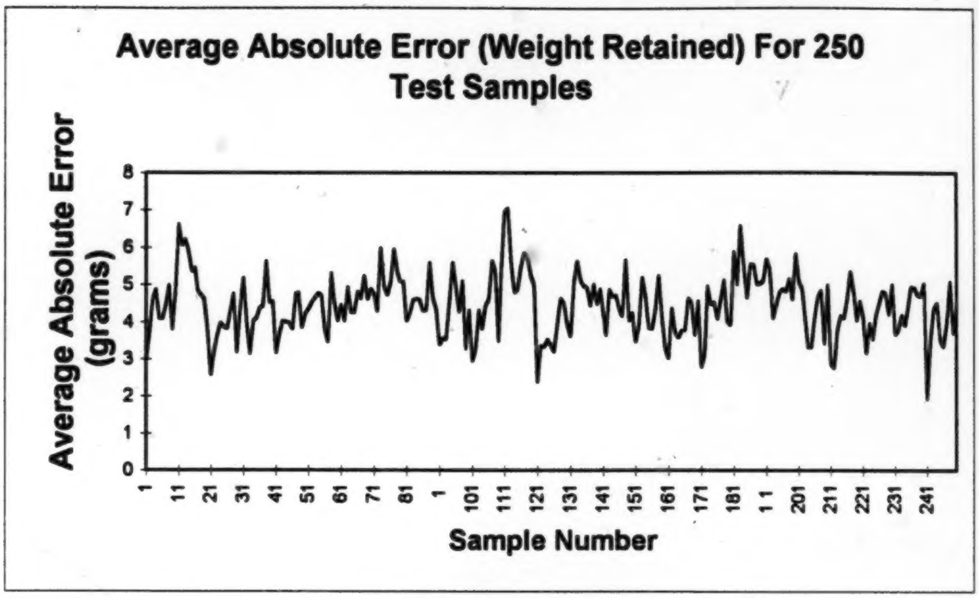


Figure 5-1 Prototype Neural Network Accuracy

5.3 Crushing Neural Network – Full Scale

5.3.1 Sample Development

Sets of input and output are required for the training of a neural network. To obtain several sets of input and CRUISER processed output, the Visual Basic program from the prototype was used. It was modified and enhanced to create gradation samples more representative of real data. The prototype model originated all samples beginning at a sieve size of 20 inches and ending at a sieve size of 0 inches. This reference line is referred to as "a gradation gradient" upon which a number of randomly created gradations are based. Adjusting the gradation gradient creates a greater variety of data. The gradient can be adjusted along the bottom size while keeping the top sieve size constant. After adjusting all bottom sieve sizes and generating the desired samples, the top sieve size is increased by one size and the bottom sieve sizes are varied again. Figure

5-2 shows a sample for each gradation gradient created when varying the bottom sieve size and keeping the top sieve size constant. A wider variety of gradation samples can be obtained this way to mimic actual samples, as discussed in chapter 2. The program operates just like the prototype in how it stores and processes the data. The neural network is then trained and tested to evaluate its accuracy. The code of the program can be found in Appendix Q. For a full-scale model, a total of 20 random gradation samples for each of the 651 selected gradation gradients were put through the program. This yielded a total of 130,200 unique training samples.

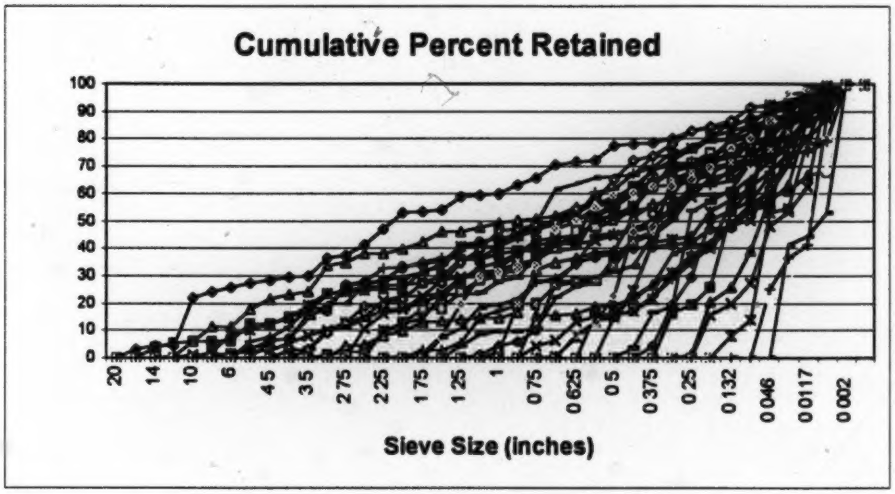


Figure 5-2 Sample Gradation Gradient Lines

5.3.2 Neural Network Architecture

The same architecture was utilized as in the prototype model. There were 41 input nodes and 40 output nodes. Of the input nodes, 40 were for the sieve sizes and 1 was for the crusher setting.

5.3.3 Preliminary Results

When the network was trained it failed during testing despite several options and attempts to improve accuracy. The researcher then decided to use a small scale sample set to determine which network variables would allow for the most accurate training. Also, it was considered feasible to evaluate the reduction in the amount of data the neural network was trying to train.

5.4 Crushing Neural Network – Trials

5.4.1 Decreasing the Number of Gradients

A sample set of 20 over 5 gradation gradients was created and put through 10 crusher settings for a total of 1000 unique samples. 900 samples were used for training the network and 100 were used for testing purposes. Varying the training parameters did very little to increase the predictive accuracy of the network. It was discovered that using only 20 samples per gradation gradient as compared to 500 samples in the prototype had some bearing on the accuracy of the model. Of the 41 input nodes in the network, 1 input is setup for the crusher setting and the remaining 40 are for the sieve sizes. The result of reducing the number of gradients from 651 to 20 was that the average absolute error reduced significantly from that of the full-scale model. However, compared to the prototype model, there was still a significant difference in average absolute error. The prototype model had an average absolute error of approximately 5 grams while the error obtained here varied from 10 to 20 grams. The average of the error for this network was approximately 14 grams per sieve size. This translates into 14 grams X 40 sieve sizes for a total of 560 grams of error for any given sample of approximately 2500 grams. This network has an error range of up to 22%. The results from this test can be seen in Figure 5-3 in section 5.4.4 under Trial #1.

5.4.2 Emphasizing the Crusher Setting

It was thought that the network would predict more accurately by placing more emphasis on the crusher setting. The crusher setting value was converted into a binary number consisting of four digits. These four digits were placed into four separate input nodes representing the crusher setting. This created less consistent predictability with little additional accuracy. The average absolute error of this network varied from 10 to 25% with some errors as high as 40%. The results from this test can be seen in Figure 5-3 in section 5.4.4 under Trial #2.

5.4.3 Cumulative Percent Retained

An idea to convert the gradation data into a form that may assist in training a neural network resulted in two possibilities: The first one was to convert the weight retained on each sieve size to a cumulative percent-retained value. The second idea was to convert the weight retained on each sieve size to a percent retained on each sieve size. The first test resulted in a network accuracy which was better than all previous attempts. The average absolute error of the weight retained on each sieve size was approximately 8 grams. The main limitation of this method is that the data was taken from only one gradient. The results from this test can be seen in Figure 5-3 in section 5.4.4 under Trial #3.

5.4.4 Percent Retained on Each Sieve Size

The second test in which the gradation data was converted into the form of a ratio resulted in a network accuracy which was even better than the prototype model. The data was converted into a percent-retained value for each sieve size instead of a cumulative retained value. The data was taken from only one gradient for this test. The researcher decided to implement the data in this ratio form on a larger scale by modelling more gradients. This would help the neural network reflect actual data more accurately. The results from the test using a single gradient can be seen in Figure 5-3 under Trial #4.

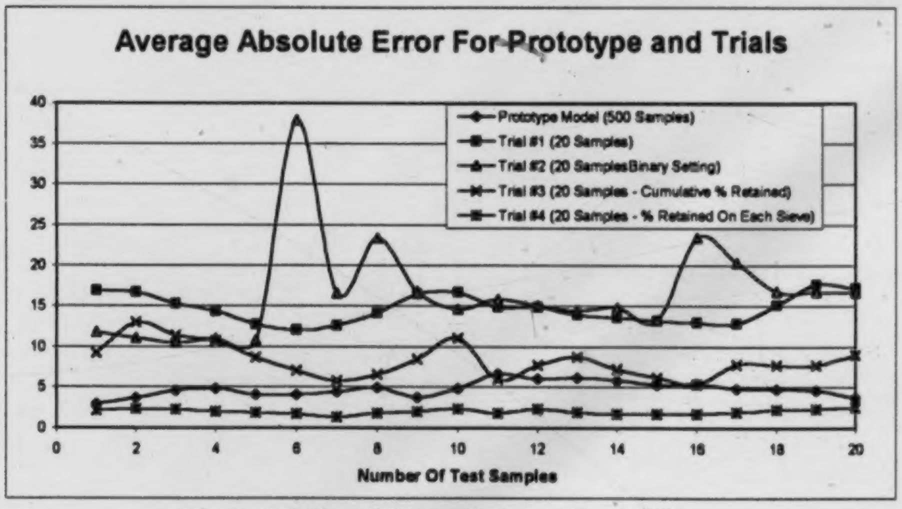


Figure 5-3 Neural Network Accuracy - Trials

5.4.5 Increasing the Number of Gradients

To enhance the ability of the neural network to effectively handle actual gradation data, the researcher decided to increase the number of gradients. It was expected that a decrease in network error would result. The main objective was to minimize the error of the network as much as possible. The previous tests all had 20 samples over all 10 crusher settings and along only one gradient. The researcher decided to remove the crusher setting node in order to reduce the error within range of a single gradient network. In order to be used in CRUISER, a network with this structure would be needed for each crusher setting. Figure 5-4 shows the network accuracy of a neural network for one particular crusher setting and 30 gradation lines. The gradient lines were chosen around gradients from actual data for a crusher with a 1 inch crusher setting. The 30 lines come from varying the starting position and finishing position of each gradient over five bottom sieve sizes and six top sieve sizes. This model was compared to a previously developed network for which there was 10 crusher settings and only 1 gradient. The prototype model results are shown to demonstrate the accuracy of the latest model, even though the results are quite high for certain test samples. All of these networks had 20 sample gradations for each gradient.

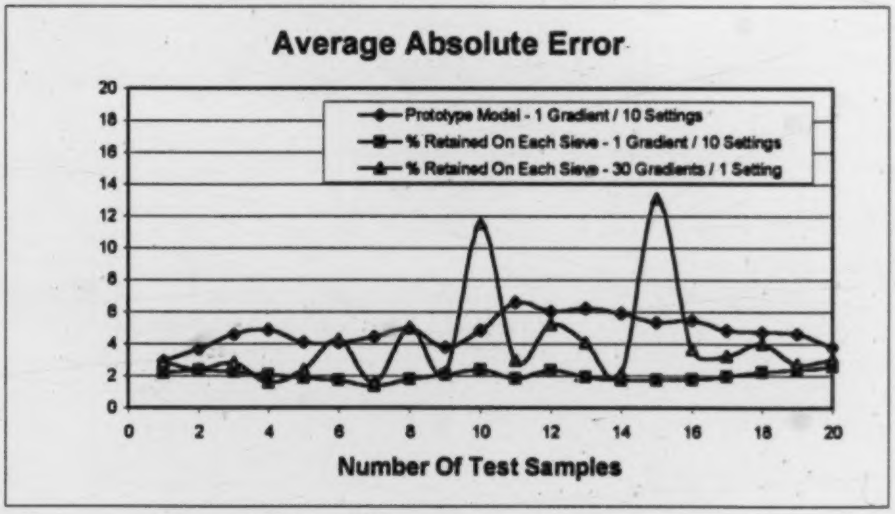


Figure 5-4 Neural Network Accuracy - Comparison

The average of the absolute error for the prototype model is 4.8; for the 1 gradient percent-retained model it is 2.0, and for the 30 gradient percent-retained model it is 4.8. The major difference between the 30 gradient model and the other two models are occasional poor test results. This is acceptable since the network represents more meaningful gradation data for the a particular crusher setting.

5.5 Crushing Neural Network – Final Model

The number of gradients within the model was optimized to best represent actual gradation data while retaining as much network accuracy as possible. It was found that the optimum number of gradients was 15, of which there was an average of the average absolute error of 3.1%. This is lower than the 30 gradient model by 1.7% and higher than the 1 gradient model by 1.1%. Using a 2500 gram sample, this translates into an error in weight of 5% as compared to 7.7% using the 30 gradient model and 3.2% using the 1 gradient model. This 15 gradient model will have a range in error of 3.8 to 8.3% using the average error and the largest and smallest sample sizes found in the data. Taking into account the odd test result of 12.5 average absolute error, the error in weight of the sample will be only 20%. This is acceptable in order to achieve a more representative neural network model to predict crushing gradations. The network accuracy of the final model, with 15 gradients, as well as the 1 and 30 gradient models are found in Figure 5-5.

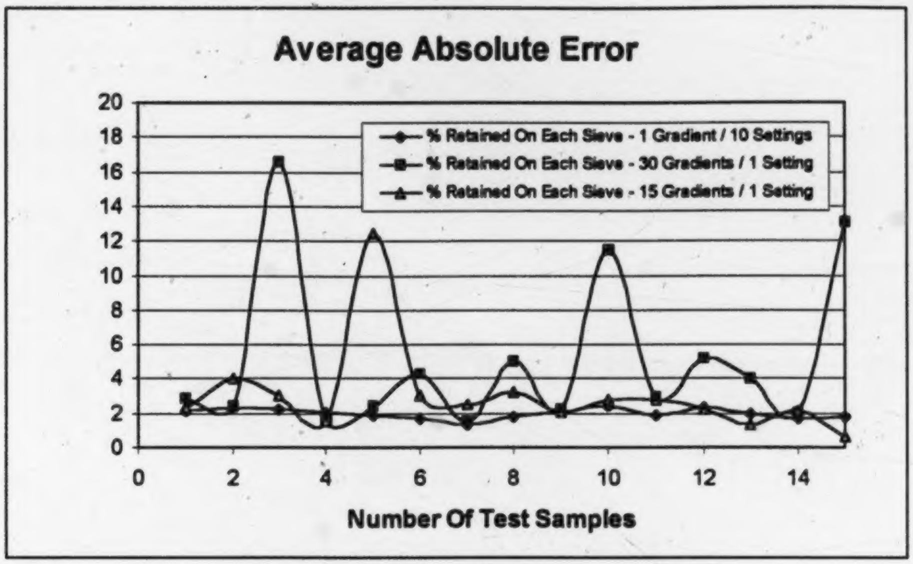


Figure 5-5 Neural Network Accuracy – Final Model

5.5.1 Neural Network Architecture

The neural network used was a feed-forward, back-propagation 3-layered neural network. The neural network architecture consists of 40 input nodes, 30 hidden layer nodes, and 40 output nodes. The input nodes contain the weight retained on each of 40 sieve sizes. The output nodes are made up of the weight retained on each of the 40 sieve sizes after being processed by the "cone crusher".

5.5.2 Optimum Training Parameters

The training parameters used for the final model were Norm-Cum learning rule, the Tanh transfer function, and a learning rate of 0.4. The neural network was trained with a total of 135,000 iterations before it was certain that the network had stabilized. The error on any given node was approximately 0.001 or less.

5.5.3 Neural Network Weights

The network weights for this network model can be found in Appendix R.

5.5.4 Model Implementation

The above mentioned model can be utilized with knowledge of the neural network weights obtained from training. These weights can be hard coded into the CRUISER program for the particular crusher type and setting. In addition, the raw gradation data can be input into a neural network trainer. Using a network trainer would allow additional data (i.e. a user's own crushing data) to be added to the existing data and retraining of the network could take place. It was found that by adding existing crushing data to the network already created through this research, the overall error of the entire network increased. This is mostly due to the randomness of the created data, which does not describe actual data in its entirety. Usually the bottom and more often the top end of the gradation curves are a lot more gradual than described by the simulated data. For the final developed model, two samples were incorporated with the created data and one sample was retained for testing. The average absolute error of this network was 7.5 grams for each sieve size. This translates into an average 12% error, whereas the final model had an average error of only 5%. The average absolute error for the added test sample was 16 grams, which translates into an error of 25%. One thing to note is that the two samples added to the model made up less than 1% of the entire sample set on which the neural network was trained. With time and the incorporation of more actual data, the neural network will establish more accurate 'actual' results.

5.6 Data Creation Procedure

For the purpose of creating more data for additional crushers, the Visual Basic program (Nndata.vbp) developed in this research can be used with a few modifications depending on the number of samples the user desires. The following procedure outlines the steps for the creation of more data and subsequent training of neural networks with the accumulated data:

1. Select the Visual Basic program for the particular crusher and copy a previously used ACCESS database. This ACCESS database will have the necessary tables and queries to manipulate the data into a form that is almost ready to be used within the neural network trainer called 'Neural Works'.
2. Determine the number of samples required to adequately train and test the network.
3. Change some of the parameters within the Visual Basic Program. The total number of samples as well as the lower and upper limit sieve sizes can be changed. For example, a lower limit of 16M and an upper limit of 4 inches was used in this research. This controlled the creation of gradation gradients for the given model. The number of random samples desired for each gradation gradient must then be entered. For example, twenty samples were used for each gradient. The total number of samples is related to the upper and lower limits as chosen by the user. In this example, the total number of samples equals to 651 with one sample for each gradient. This 651 is then multiplied by the number of random samples desired for each gradation gradient (i.e. 20) for a total of 13,020 unique

samples. Each of these samples is put through all possible crusher settings (i.e. 10) for a total of 130,200 unique input and output samples for the neural network. The reference to the ACCESS database within the program should be changed to the name of the copied database.

4. Run the program. This may take some time depending on the number of samples desired by the user.

5. After the program is finished, open the database where the data is stored. Within the database, a total of three queries must be run. The first query is called CombinedQuery which incorporates an InputCrosstab and OutputCrosstab queries. Using Visual Basic code to handle the data within ACCESS, the CombinedQuery puts the input and output data in a form for the neural network software. The second query is called GenNNtrainData; it extracts the randomly created samples from the table created by the CombinedQuery to be used for training the neural network. Criteria within this query can be modified to adjust the number of samples desired for training and subsequent testing. Usually the number of data sets used for testing are about 10 to 15% of the total number of data sets. The third query is called GenNNtestData; it extracts the samples which are not extracted to train the neural network but will be used to test the network after it is trained.

6. The total weight retained for each sample in the training set is then calculated by running the CalcTotalRetForNNTrainData, which creates a NNTrainDataWgtRet table and extracts data from the NNTrainData table. This is also done for the

testing set of data by running the CalcTotalRetForNNTestData, which creates a NNTestDataWgtRet table and extracts data from the NNTestData table.

7. The percent retained on each sieve size is calculated by running the NNTrainData%Ret and NNTestData%Ret queries. These queries create the NNTrainDataPerRet and the NNTestDataPerRet tables. These tables contain the data in the appropriate form for the training and test of the neural network.

8. The sample numbers must be deleted from the NntrainData%Ret table and the NntestData%Ret table. Each of these tables must be selected separately and saved as a text file in space delimited format in order to load the data into the neural network software. If the headings are retained in the text file then a "!" is to be placed in front of the text row in the text file. The file extensions for the training and testing files need to be changed from "filename.TXT" to "filename.NNA" using Windows Explorer.

9. Open the neural network training program, Neural Works, and change the directory to where the two files are located. Select the training and testing files from within the program and input the number of input and output nodes required by the data, as well as the number of hidden layer nodes desired. The neural network can now be trained and tested. The output from testing the neural network can be found in the file "filename.NNR" This file contains the data sets used for testing and the predicted sets of data from the trained network.

10. Evaluate the accuracy of the network by comparing the actual output data to the predicted output data.

5.7 Implementation of Neural Networks

A neural network trainer is to be developed for a number of projects within the Construction Engineering and Management program at the University of Alberta. This network trainer will be incorporated into CRUISER to enhance the modeling features of the program. The neural network parameters developed in this research could be hard coded into the CRUISER program. The developed trainer could also use the created data to train a network within the CRUISER program. The interface for the crusher option within CRUISER will be modified to allow the user to choose the desired crusher analysis method. The first option would be the old method of analysis, in which the output gradation is selected from a chart depending on the crusher setting. The second option would be to use a developed neural network for a particular type of crusher. This could be either hard coded or redeveloped using the developed neural network trainer. A third option would be to choose a neural network developed by the user. A neural network could be created by putting input and output data into a blank spreadsheet and then training and testing the network.

5.8 Conclusion

The results of this chapter clearly indicate that the developed neural network model will serve its purpose to expand the crushing analysis capabilities of CRUISER. This research will facilitate the use of neural networks and the incorporation of actual data by the users of the program. Additional work to create neural networks for additional crusher types and settings must be developed. The collection of actual data will allow for more accurate modeling but is not absolutely necessary. One option could be to develop neural networks for common crusher types and settings. Other types could be handled by supplying a neural network framework to incorporate the user's own crushing data. This would be done through the use of a neural network trainer. A neural network trainer within the CRUISER program will allow a user to add additional data to an existing network, retrain the network, and use the developed crushing network. The user could select from three options when configuring a crusher within CRUISER. One option is to use gradation data from an aggregate production handbook. A second option is to select the analysis to use an already trained neural network. The third option would be to add additional data to an existing network and retrain it before using it for analysis. It is recommended that users develop their own data for the crusher types and settings they use most often. This would result in more accurate results than just adding a few sets of data and retraining an already developed network. Obtaining data and developing a network from scratch will allow for the neural network to become more aggregate pit, equipment, and product specific. This will in turn create a more accurate and meaningful crushing analysis for the user of the CRUISER program. Adding accuracy to the program will increase its acceptability and reliability, thus encouraging aggregate production personnel to use the program.

6.0 Optimizing Parameter Selection Using Belief Networks

6.1 Introduction

In general, the initial parameters of the CRUISER program will be setup by the user and then go through one simulation process. After this simulation is complete and the resulting output gradation is viewed, the user will have the option to say if he/she would like a suggestion to bring the output gradation within specifications. Choosing this option will then prompt the user to tell the model what to optimize. The model will then optimize the specifications. Once the model knows this, it will look at the probabilities associated with the elements of the model and how they affect the gradation. Evaluating the probabilities, CRUISER will then suggest a change to the model. The user can then make this change to the model and run the simulation for the second time. Presently, the changes to the model are done manually by the user; this is an acceptable means of model optimization. In future work the link between the belief network model and CRUISER can be accomplished by a computer programmer. This will, in effect, make CRUISER a 'smart simulation' model. If desired, the model developed in this research can be incorporated into the CRUISER program in an automated fashion.

The first characteristic of belief networks that made it a strong candidate for diagnostic purposes was the ability to approach the problem from more than one direction. Although the model developed in this research does not take full advantage of this benefit, the network can be expanded at any time. Either the known states of variables could be entered as evidence, or the diagnostic tool could be permitted to determine the likelihood of a variable being the cause of poor performance in the absence of evidence.

This ability of adjusting variables to be input or output without having to redesign the system is uncommon to other forms of artificial intelligence. For example, neural networks do not allow for different input variables from one simulation to another. Instead, the model is trained with specific inputs and outputs and can only predict adequately when given these inputs. This research utilizes belief networks to allow a number of different combinations of input into the network and giving relevant output. The greater amount of input evidence provided to the network will exploit the network's ability to combine probabilities to show the most likely causes for the numerous observed pieces of evidence. Another strong characteristic of belief networks is the ability of the model to accept expert opinion instead of requiring historical data, which is not always available, nor in desired quantity and quality.

6.2 Network Development

The development of the belief network consisted of combining expert advice and computer simulation routines to arrive at a relatively generic network. The first phase of development began with obtaining information from an expert in the field of aggregate production. This information was the basis of more than half of the nodes that comprise the model. The second stage involved a sensitivity analysis of the CRUISER program and discovery of the degree to which characteristics affected the product gradation. This was done to enhance the network by including more features of the simulation program, therefore making the program more educational. The features of the simulation model that were to be added to the model were ranked by both the researcher and Bill Laisse according to the importance and relevance of the factor in changing the gradation. The expert advice and ranking order was then combined into the model in the form of probabilities for each node. After attaching the probabilities to each factor, a testing phase was implemented to evaluate the accuracy of the network in assessing the causes of the product being out of specifications with a given layout. Not only was the model to be tested with the plant layout from which it was developed, but also with smaller models, which had fewer pieces of equipment than the original one. The model on which the belief network was modeled is shown in Figure 6-1. This model was chosen primarily because the expert and the researcher were most familiar with its corresponding plant layout. Developing a model of larger magnitude would involve an additional industry contact and a more complicated aggregate plant site. The location of the plant layout used for this research was only 1 hour from the city of Edmonton, which made discussions with plant personnel and progress of the research much more efficient.

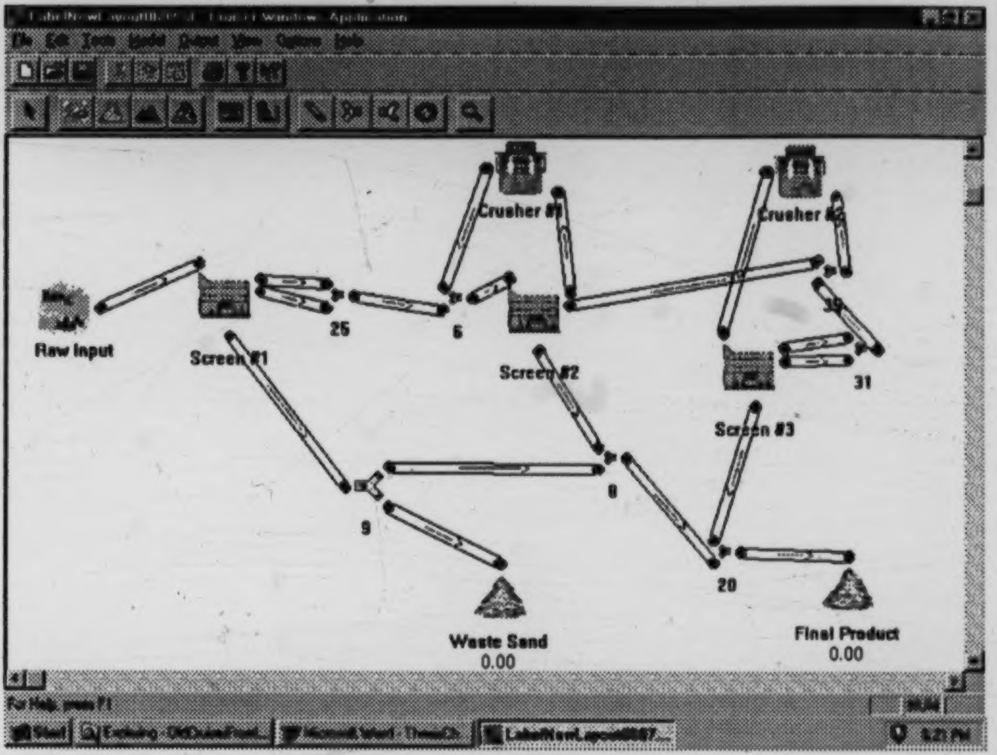

Crusher #1
Crusher #2
Raw Input
Screen #1
25
6
Screen #2
30
31
Screen #3
8
9
20
Waste Sand
0.00
Final Product
0.00
For Help, press F1

Figure 6-1 – Initial Plant Layout

The researcher decided not to utilize the ability of belief networks to suggest to the user the probability of any one causal node being responsible for an inadequate product in the absence of evidence would not be utilized. It was not deemed to be a useful tool for this particular application, since each causal node would have different probabilities based on the equipment used and the initial characteristics of the plant. For a less experienced user at the plant setup stage there would likely be an equal probability of any one cause resulting in an inadequate product. An inexperienced user is likely to begin with a number of plant characteristics that will require changing. For a more experienced user, the initial characteristics of the plant will be closer to the final characteristics that are required to produce a product within specifications. In this case, the probabilities of any one causal node being responsible for making an inadequate product may be reflected differently because of the specific equipment being used and the experience of the plant superintendent. The experience of the superintendent will affect the probabilities of the causal nodes because each superintendent will assess the probabilities differently, depending on the characteristics of the plant at the start of plant testing. When no evidence is given to the belief network, varying types of equipment and levels of superintendent experience make it more difficult to model the probabilities of the causal nodes. Belief networks will only be useful as an optimization tool if evidence is provided to the network to evaluate the causes or vice versa. Since it is easier to attain the evidence observed rather than the resulting cause, this is the usual starting point. For curiosity's sake, it may be interesting to determine the probabilities of any one node being the cause; however, this would be meaningless for optimizing the characteristics of the aggregate plant. It is very difficult to obtain the optimal solution if one only knows one side of the equation and not the result. For this to be possible, a plant superintendent would have to know the exact characteristics of the plant from the very beginning, and experimentation would no longer be necessary.

6.3 Expert Information

Much of the expert information on how an aggregate production plant is run and operated was obtained from Bill Laisse of Lafarge. Questions were asked and the pertinent information gained from these visits was recorded to develop the belief network for practical use. Some of the significant factors were the screen sizes and shapes at particular locations in the plant layout, crusher settings, and regulating native sand input. Some actual diagnostics while operating an aggregate plant were also included in the model. This will give some insight to users who lack hands-on plant experience, telling them what factors to look for to source the problem resulting in an inadequate product.

A number of on-site factors had to be assessed as to their practicality for inclusion in the model. Changing the bowl and mantle of the crushers is considered to be a maintenance item; in practice they are checked more often when it seems likely that they are wearing out. This can vary from 3 weeks to 6 weeks, depending on the equipment used and how hard the crusher has to work. Since this varies so much from site to site, it was left out of the model. Some of the diagnostics of the model could only be interpreted as approximate since the relationship between certain variables will vary from one pit to another. For instance, how changing the operating speed of a screen will affect the gradation of the final product depends on the equipment. In practice, aggregate producers use various screens that are composed of light or heavy wire. A screen with lighter wire will provide more hole openings over the full deck of the screen. This will allow for a slight increase in production and will generate a coarser product. The typical range of increasing or decreasing the surface area opening on a plant site is ± 5%. The options within CRUISER allow for up to ± 20%. Although the full range of this option within CRUISER is not used in the practical field, it was still a useful option for enhancing the simulation model. Since adjusting the surface area of the screen will indeed affect the gradation to a small degree, this option was included within the belief network model.

One problem with testing the belief network against the CRUISER program is that the inaccuracy of the program is quite sensitive to the crusher setting. As discussed previously, this is because the program presently describes the output of a crusher at a particular setting which remains the same no matter what the input gradation is. So the setting for secondary or finishing crushers in the plant will have far greater affect than primary crushers on the product gradation. This is also true in the real world, but is not totally characteristic of what actually happens. Just the same, the belief network was designed to best represent the actual situation within the confines of the CRUISER program. It is believed that the network will perform better in this respect once the program has enough input into the developed neural networks to model crushers more effectively.

Crusher settings can be adjusted from one to three times daily both to keep the material flow optimum to the crushers and to keep the final product within specifications. It is also common practice to observe these material flow changes while the aggregate plant is in operation. The way the settings should be adjusted is determined through a trial and error approach based on experience and specific conditions. It is also common practice to inspect the screens on the top decks for holes at regular intervals since little evidence of holes on these screens is found by monitoring the final product gradation. This is because oversized material that does pass through the screen will still proceed through a crusher instead of arriving at the final product. Changing the speed of a screen can also affect the product gradation. Increasing the speed of a finishing screen will cause more material that could pass through the screen to continue on through the crushing process again. This will result in a finer product. The opposite is true for the same reasons:

Decreasing the speed will cause some material that is likely to go through the crushing process to pass through the screen and appear in the final product. It was found that the pitch of a screen (angle at which the material is thrown) was selected for a plant with consideration of the equipment being used and material being processed. For these reasons, the degree of pitch was left out of the belief network. It was also revealed through discussions that cross-slot screens are only used on the scalper screen, where sand screening is required. CRUISER contains options for implementing slotted screens on any deck of any screen; the model is therefore focused around the program for the factor of slotted screens. The maximum slot length/width ratio of a screen size was incorporated into the model but must be used with caution because it varies in actual practice as to what the specifications allow from job to job. It is assumed that the user is conforming to the specifications of the job both when using this option within CRUISER and when evaluating it within the belief network. Another on-site occurrence is when material clumps up and either restricts material flow or ends up in the final product in a smaller clump. The clumping of material occurs during the winter months due to sub-zero temperatures and can increase the amount of fines < 5mm in the final product. Bill Laisse also informed the researcher that there are two factors that affect the % fracture in the final product: One factor is the amount of sand in the final product; the other is the use of screens smaller than the product top size on the bottom deck of the finishing screen. The second factor will cause more material to be sent through the finishing crusher, resulting in a slightly finer product. The first solution for a % fracture problem, is to change the amount of sand in the product if the specifications allow for it. If the specifications restrict this, then some screens on the bottom deck of the finishing screen are made smaller or larger, depending on the top size of the final product. Sometimes there are two causes for the same result with an equal probability of occurring, but one cause is easier and more likely to be corrected in the real situation. In this case, the researcher attaches a greater probability to the cause that is most likely to be changed in the real situation. If this cause cannot be changed, then the user should progress on to the second suggestion from the network, continuing with any one suggestion as long as the user is satisfied with the improvements to the product gradation. For example, if the final product has too high a percentage of fractures, there are two possible causes: One cause is that there is too little sand in the product, and the other is that the splits on the further end of the material flow on the bottom deck of the finishing screen are too small below the maximum top size of the final product. Each cause has an equal chance of occurring, but the more likely resolution to this problem would be to increase the amount of sand to the product since this is the quickest and easiest to do in the field. However, specifications may prohibit adding additional sand to the product. In this case, the second option, that of changing the screens should be used to resolve this problem.

The belief network is designed for a plant layout that produces one final product. Therefore it must be noted that if any intermediate or secondary products are being made, it is up to the user to make logical plant changes to attain the specifications for these intermediate products. However, it has been found that the belief network can still be used as an aid in making corrective decisions for smaller plant layouts. This will be discussed further in the testing section of the belief network model.

Another assumption of the belief network model is that to gain maximum production from a plant, the undersize of the bottom decks of any primary or intermediate screen will proceed to the final product without any additional processing. This is typically observed in the field and is a logical step in maximizing production. It was discovered that the belief network could not account for some of these logical decisions in attaining increased production or an acceptable gradation.

In actual practice, a jaw crusher is sometimes used in place of a scalper screen. Use of a jaw crusher would increase production, but only slightly. The jaw crusher would have a grizzly before it to screen out the sand before the crushing process. Gates to regulate the flow of sand would be found below the grizzly and would act in the same manner as those found in the scalper screen. The only other change would be that the secondary crusher could have a medium head bowl instead of a coarse head bowl. The brunt of the crushing process would still be with the third, or finishing, crusher. The net effect of using a jaw crusher in place of a scalper screen on the final product gradation is negligible, so this was left out of the model. The setting on a jaw crusher that proceeds two other crushers is changed to regulate the material flow in the system, and not to correct any changes in the final product gradation.

6.4 Simulation Information

As previously discussed, some additional factors were implemented into the model after the expert opinion was implemented. Some of these factors would only make slight adjustments to the gradation output of the simulation model but were incorporated into the belief network to further optimize the gradation, even to a small degree. A sensitivity analysis was performed on the CRUISER program to evaluate the changes affecting the gradation for two areas, one being >5mm and the other being <5mm. This division is used in practice as the basis for determining plant adjustments and changes. It divides the product into two relatively equal parts, as shown in Figure 6-2. The product gradation in this figure is for a 15mm aggregate. The dashed lines are the high and low boundary specifications. If the product gradation line is above the upper boundary, the gradation is too fine. Likewise, if it below the lower boundary line, the gradation is too coarse. The solid line is the simulated product gradation from the CRUISER program. Following the analysis, factors were ranked according to their importance in affecting the gradation either positively (i.e., finer) or negatively (i.e., coarser) on a graph, which shows the results in a percent passing format. These factors were combined with the factors provided by Bill Laisse to arrive at the variables included in the network model. The probabilities for each factor were determined through discussions with Bill Laisse to ensure that the model would reflect the actual causal probabilities.

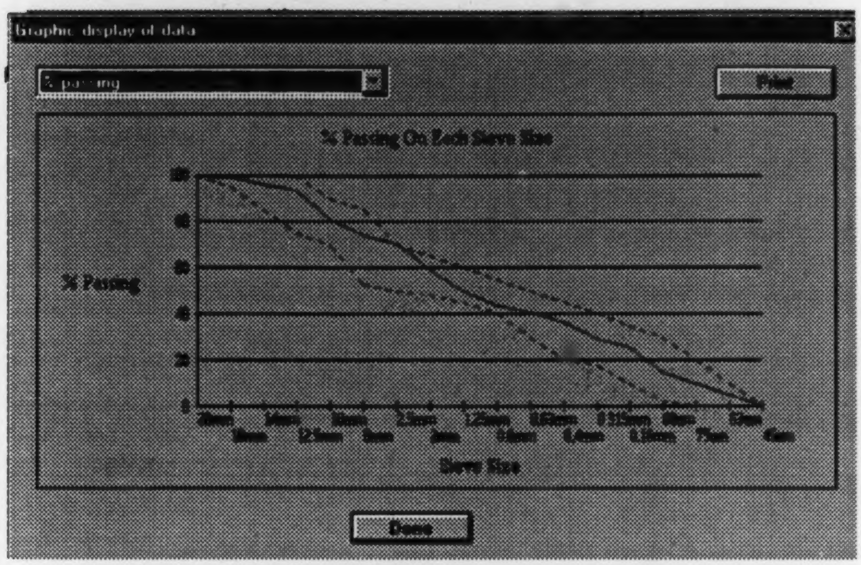

Graphic display of data
% passing
Print
% Passing On Each Sieve Size
% Passing
100
80
60
40
20
0
12.5mm
2.5mm
1.25mm
0.315mm
Sieve Size
Done

Figure 6-2 – Output Gradation as Observed in CRUISER

6.5 Belief Network Structure and Probabilities

The network is made up of seven evidence nodes and thirty-one causal nodes. Of the thirty-one causal nodes, twenty-five are directly connected to the evidence nodes, which will be referred to as 'first layer causal nodes'. The other six causal nodes are indirectly connected to the evidence nodes through the twenty-five directly connected nodes. These nodes will be called 'second layer causal nodes'. The entire network as viewed in Microsoft Belief Networks (MSBN) is shown in Figure 6-3.

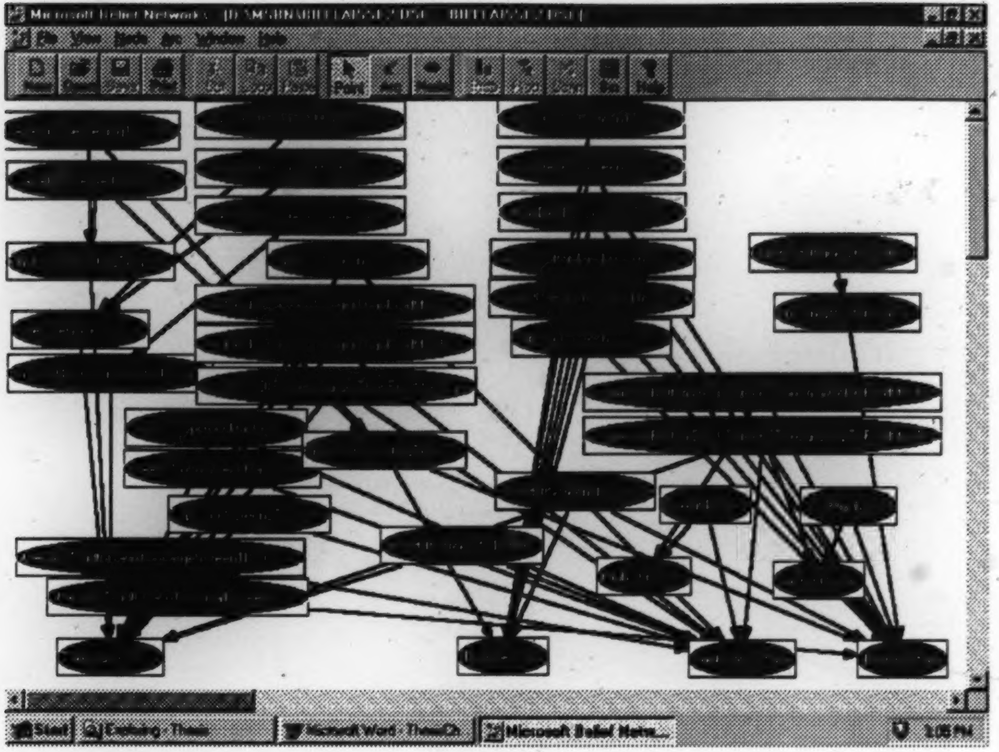


Figure 6-3 – Aggregate Production Belief Network

Of the seven evidence nodes, four of them will be called primary evidence nodes since they pertain to the gradation parameters with which the researcher is most concerned. These four nodes are located at the bottom of the network, as shown in Figure 6-3. They are labeled: “Too Coarse >5mm”, “Too Coarse <5mm”, “Too Fine >5mm”, and “Too Fine <5mm”. Two other evidence nodes are linked to another specification of the aggregate product; these can be considered to be secondary evidence nodes. These nodes refer to the percentage of aggregate that has been fractured during the crushing process. These two nodes are: “High % Fracture” and “Low % Fracture”. Another evidence node, which accounts for a common diagnostic problem during the winter months, is considered to be a tertiary evidence node. This node is labelled "Presently Winter Season". Over the winter months sub-zero temperatures can cause fines to clump and, as a result, the total amount of fines in the final product increase.

The following table lists the description of selected causal nodes for interpretation by a user of the belief network model. The corrective action the program user should make to the plant characteristics is also given.

Table 6-1 Causal Node Descriptions and Corresponding Corrective Actions

Causal Node Description	Corrective Action
Coarse Crusher Setting Too Large	Decrease the coarse crusher setting.
Screen1 Bottom Deck Possibly Damaged	Replace the damaged screen at the described location.
Top Deck Screen3 Too Large	Decrease the screen size of a split or deck at the described location.
Bottom Deck of Screen3 Larger Than Prod Max Top Size	Ensure the screen size is no greater than the maximum size of the product and decrease the screen size of a split or deck at the described location.
Screen3 Bottom Deck Has Light Wire	Use a coarser type of wire for the entire deck.
Increase Slot Length/Width Screen1 Bottom Deck	Increase the Slot Length/Width ratio of a split or deck at the described location.
Speed of Screen3 Is Too Fast	Decrease the speed of screen 3.
Clumping of Fines Due To Frost	Try to obtain the raw product from a more localized area to reduce the amount of frozen soil entering the plant.
Further Splits On Bottom Deck Screen3 Too Small Below Prod Max Top Size	Increase the screen size for splits furthest away from where the material initially flows onto the screen at the described location.
Too Little Sand	Increase the flow of native sand to the final product.

The following seven tables show the probabilities that were entered for each causal node. Only the four primary nodes have both first and second layer causal nodes. Second layer causal nodes will be described as being connected to first layer causal nodes or "sub-causal" nodes. Remember that the probabilities in a belief network reflect the probability that the causal node is false. In other words, the lowest probability is the causal node most likely to be responsible for the present result. The probabilities were generated by way of the following process. For each evidence node, the corresponding characteristic and diagnostic nodes were listed in a ranking order. This order was determined through discussions between the aggregate production expert and the researcher. Some characteristic probabilities were determined with more influence by the researcher to whom the operation of the CRUISER program was more familiar. Some characteristic probabilities were not of much interest to the expert and these probabilities were again determined by the researcher. These probabilities reflect the factors to which the CRUISER program is more or less sensitive in optimizing a product gradation

Table 6-2 Probabilities for Too Coarse > 5mm Causal Node – Layer #1

Probability	**Causal Node Description**
0.1	Bottom Deck of Screen 2 is Too Large
0.3	Bottom Deck of Screen 3 is Too Large
0.4	Oversize Rock Is Cracked or Fractured
0.4	Screen 3 Bottom Deck Has Light Wire
0.5	Screen 3 Top Deck Has Light Wire
0.55	Top Deck of Screen 3 is Too Large
0.55	Fine Crusher Setting is Too Large
0.6	Oversize Rock is Round
0.8	Top Deck of Screen 2 is Too Large
0.8	Further Splits on Bottom Deck of Screen 3 are Too Large up to the Maximum Top Size of the Product
0.9	Bottom Deck of Screen 1 is Too Large
0.9	Speed of Screen 3 is Too Slow

Table 6-3 Probabilities for Too Coarse > 5mm Causal Node – Layer #2

Sub Causal Node Description	Probability	Causal Node Description
Oversize Rock is Cracked or Fractured	0.4	Screen1 Bottom Deck May Be Damaged
Oversize Rock is Cracked or Fractured	0.6	Screen 2 Bottom Deck May Be Damaged
Oversize Rock is Round	0.9	Screen 3 Bottom Deck May Be Damaged
Fine Crusher Setting is Too Large	0.3	Coarse Crusher Setting is Too Large
Fine Crusher Setting is Too Large	0.7	Coarse Crusher Setting is Too Small

Table 6-4 Probabilities for Too Coarse <5mm Causal Node – Layer #1

Probability	Causal Node Description
0.1	Bottom Deck of Screen 2 is Too Large
0.3	Bottom Deck of Screen 3 is Too Large
0.3	Screen 3 Bottom Deck Has Light Wire
0.5	Too Little Sand
0.6	Screen 3 Top Deck has Light Wire
0.6	Top Deck Screen 3 is Too Large
0.6	Slot Length/Width Ratio of Screen 1 Bottom Deck is Too Large
0.8	Further Splits on Bottom Deck Screen 3 are Too Large
0.8	Top Deck Screen 2 is Too Large
0.9	Speed of Screen 3 is Too Slow

Table 6-5 Probabilities for Too Fine >5mm Causal Node – Layer #1

Probability	Causal Node Description
0.1	Bottom Deck of Screen 2 is Too Small
0.3	Fine Crusher Setting is Too Small
0.4	Bottom Deck of Screen 3 is Too Small
0.5	Screen 2 Bottom Deck Has Heavy Wire
0.5	Screen 3 Bottom Deck Has Heavy Wire
0.7	Top Deck Screen 3 is Too Small
0.8	Bottom Deck of Screen 1 is Too Small
0.9	Speed of Screen 3 is Too Fast

Table 6-6 Probabilities for Too Fine > 5mm Causal Node – Layer #2

Sub Causal Node Description	Probability	Causal Node Description
Fine Crusher Setting is Too Small	0.7	Coarse Crusher Setting is Too Large
Fine Crusher Setting is Too Small	0.3	Coarse Crusher Setting is Too Small

Table 6-7 Probabilities for Too Fine < 5mm Causal Node – Layer #1

Probability	Causal Node Description
0.1	Bottom Deck of Screen 2 is Too Small
0.3	Too Much Sand
0.3	Screen 2 Bottom Deck Has Heavy Wire
0.5	Screen 3 Bottom Deck Has Heavy Wire
0.6	Fine Crusher Setting is Too Small
0.6	Top Deck of Screen 3 is Too Small
0.8	Bottom Deck of Screen 3 is Too Small
0.8	Further Splits on Bottom Deck of Screen 3 are Too Small
0.8	Slot Length/Width Ratio of Screen 1 Bottom Deck is Too Small
0.9	Speed of Screen 3 is Too Fast

Table 6-8 Probabilities for Too Fine < 5mm Causal Node – Layer #2

Sub Causal Node Description	Probability	Causal Node Description
Fine Crusher Setting is Too Small	0.7	Coarse Crusher Setting is Too Large
Fine Crusher Setting is Too Small	0.3	Coarse Crusher Setting is Too Small
Presently Winter Season	0.95	Clumping of Fines Due to Frost

Table 6-9 Probabilities for High % Fracture Causal Node

Probability	Causal Node Description
0.6	Further Splits on Bottom Deck of Screen 3 are Too Small Below the Maximum Top Size of the Product
0.4	Too Little Sand

Table 6-10 Probabilities for Low % Fracture Causal Node

Probability	Causal Node Description
0.6	Further Splits on Bottom Deck of Screen 3 are Too Large up to the Maximum Top Size of the Product
0.4	Too Much Sand

There is a twenty percent chance of either a high or low percentage of fractures occurring and neither causal node being responsible. This twenty percent chance of other causes is mostly due to the nature of the material, which can be highly variable.

Other probabilities with small and seemingly insignificant values had to be entered so that the belief network would work. The network requires probabilities greater than zero for each causal node in order for it to calculate combined probabilities when more than one node is given evidence. For example, evidence nodes can have a probability of occurring due to a cause not included in the network. This cause could be unknown or not included because of its complexity. The causal nodes in the model must also have causal probabilities when no evidence is provided to the network. This was difficult to evaluate because it is contingent on how the plant is laid out and the predetermined characteristics of the model for the first simulation run. The probability assigned to each causal node is 0.5 when no evidence is provided to the network. This value was chosen arbitrarily as a starting point. With all nodes having the same value and no evidence being given to the network, each node has an approximately equal chance of being the cause for the gradation to be inadequate when compared to the specifications. Therefore, with no evidence given to the network the probabilities should be around 0.5. The only probabilities that will not be exactly 0.5 are those causal nodes which connect other causal nodes indirectly to the evidence nodes.

To understand the power of combining probabilities via the belief network, the network will be presented with evidence supplied first to only one node and then to two nodes. Before doing this, some heuristic rules for which a user of the network model should follow will be presented. Some more specific rules will be discussed later on in the section on using the belief network. It was decided that a set of heuristics would be added to the model to allow for three different scenarios. The first scenario is when a person with very little aggregate plant production experience is using the model along with the CRUISER program. This scenario could occur in an educational setting, for example. When using the belief network model in this situation, the user should only use suggestions from the network pertaining to the CRUISER program (i.e., characteristic suggestions). The second scenario is during the operation of a plant where an adequate product has been produced with the given setup. When using the belief network model in this situation, the user should only use the diagnostic suggestions (i.e. expert knowledge) from the network. These suggestions would pertain to a change in gradation due to slight changes in the plant that may periodically require attention. Suggestions to change the plant characteristics are not usually necessary at this stage, since the plant has previously produced a satisfactory product and only minor equipment adjustments or component replacements are required. The user would not need to use both the CRUISER program along with the belief network for this scenario. The third scenario is during the primary setup and gradation testing of a plant. When using the belief network in this situation, the user could use both sets of suggestions or the belief network model in its entirety. In this scenario, the plant characteristics are undergoing experimentation to produce a product that meets specifications. At the same time, there is a chance that the gradation may be brought within the specifications with only minor equipment adjustments. Since changing the characteristics of the plant is more likely to bring the gradation within specifications, these nodes within the network are given higher probabilities of being the causal node than those which are more diagnostic in nature. A more experienced CRUISER user may want to utilize the model in this scenario to help educate him/herself as to what problems actual plant personnel may encounter when operating an aggregate plant. The primary evidence nodes are to be used for the first scenario. The secondary and tertiary evidence nodes are to be used for the second scenario. All evidence nodes are to be used for the third scenario.

Evidence may be supplied to the network through one node or through a combination of nodes. The case where only one node is given evidence will be shown here, even though the probabilities indicate the same ranking as what was given to the network at the time of development. Then evidence into another node will be added to the network and the changes in the causal probabilities will be recorded. Evidence into another node will be added to the first two nodes and the probabilities will be recorded again.

The connected causal nodes for the "Too Coarse > 5mm" evidence node are presented in Table 6-11. This table shows the node's probability, rank, whether the node is characteristic or diagnostic, and a description of the node. Remember that characteristic nodes are to be used for scenarios one and three. The diagnostic nodes are to be used for scenarios two and three.

Table 6-11 Case #1: Too Coarse > 5mm

Probability	Rank	Nature of Cause (C or D)	Causal Description
0.09	1	C	Bottom Deck of Screen 2 is Too Large
0.23	2	C	Bottom Deck of Screen 3 is Too Large
0.29	3	D	Oversize Rock is Cracked or Fractured
0.29	4	C	Screen 3 Bottom Deck has Light Wire
0.33	5	C	Screen 3 Top Deck has Light Wire
0.35	6	C	Top Deck of Screen 3 is Too Large
0.35	6	C	Fine Crusher Setting is Too Large
0.37	7	D	Oversize Rock is Round
0.44	8	C	Top Deck of Screen 2 is Too Large
0.44	8	D	Further Splits on Bottom Deck of Screen 3 are Too Large
0.47	9	C	Bottom Deck of Screen 1 Too Large

For the second example of output from the belief network another primary node will be given evidence. This node will be the "Too Coarse < 5mm" node. The case refers to whether the causal node is primarily linked to the first causal node, the second causal node, or both. The cause is again whether the node is characteristic or diagnostic.

Table 6-12 Case #2: Too Coarse > 5mm and Too Coarse < 5mm

Probability	Rank	Case and Cause (C1, C2, CB, D1, D2 or DB)	Causal Description
0.01	1	CB	Bottom Deck of Screen 2 is Too Large
0.08	2	CB	Bottom Deck of Screen 3 is Too Large
0.11	3	CB	Screen 3 Bottom Deck has Light Wire
0.23	4	CB	Screen 3 Top Deck has Light Wire
0.24	5	CB	Top Deck of Screen 3 is Too Large
0.29	6	D1	Oversize Rock is Cracked or Fractured
0.33	7	C2	Too Little Sand
0.35	8	D1	Oversize Rock is Round
0.37	9	C1	Fine Crusher Setting is Too Large
0.37	9	C2	Slot Length/Width of Screen 1 is Too Large
0.39	10	CB	Top Deck Screen 2 is Too Large
0.39	10	DB	Further Splits on Bottom Deck Of Screen 3 are Too Large
0.44	11	D2	Speed of Screen 3 is Too Slow
0.47	12	C1	Bottom Deck of Screen 1 is Too Large

Adding evidence to two nodes instead of one has increased the number of possible suggestions from the network and has rearranged the ranking order of the suggestions. Only the first two suggestions remain in the order they were in when evidence was supplied to only one node. The primary nodes are the evidence nodes connected to the majority of the causal nodes. Any permissible dual combination of these four evidence nodes will indeed improve the ranking order of the suggestions more than any dual combination of a primary node and secondary or tertiary node. However, adding evidence to two primary nodes and a secondary node will affect the ranking order of the causal nodes linked to the secondary node to a greater degree.

Table 6-13 Case #3: Too Coarse > 5mm and Too Coarse < 5mm and High % Fracture

Probability	Rank	Case and Cause (C1, C2, CB, D1, D2 or DB)	Causal Description
0.01	1	CB	Bottom Deck of Screen 2 is Too Large
0.08	2	CB	Bottom Deck of Screen 3 is Too Large
0.11	3	CB	Screen 3 Bottom Deck has Light Wire
0.23	4	CB	Screen 3 Top Deck has Light Wire
0.24	5	CB	Top Deck of Screen 3 is Too Large
0.29	6	D1	Oversize Rock is Cracked or Fractured
0.17	3.5	C2,3	Too Little Sand
0.35	8	D1	Oversize Rock is Round
0.37	9	C1	Fine Crusher Setting is Too Large
0.37	9	C2	Slot Length/Width of Screen 1 is Too Large
0.39	10	CB	Top Deck Screen 2 is Too Large
0.38	9.5	D1,2,3	Further Splits on Bottom Deck Of Screen 3 are Too Large
0.44	11	D2	Speed of Screen 3 is Too Slow
0.47	12	C1	Bottom Deck of Screen 1 is Too Large

Even though the ranking order only changed the two causal nodes that were connected to the secondary evidence node added, the optimization of arriving at an acceptable gradation is improved. This additional evidence node, like all other non-primary evidence nodes, is diagnostic in nature. Evidence into these nodes will only be required when the network is used in scenarios two or three as previously discussed.

The total number of possible combinations of evidence can then be supplied to the developed network is seventeen. The other fourteen combinations of evidence nodes into which the node can be stated as true are listed below:

1. Too Coarse <5mm
2. Too Coarse<5mm and High % Fracture
3. Too Fine >5mm
4. Too Fine <5mm
5. Too Fine <5mm and Low % Fracture
6. Too Coarse >5mm and Too Fine <5mm
7. Too Coarse >5mm, Too Fine <5mm and Low % Fracture
8. Too Coarse >5mm, Too Fine <5mm and Presently Winter Season
9. Too Coarse >5mm, Too Fine <5mm, Low % Fracture and Presently Winter Season
10. Too Fine >5mm and Too Coarse <5mm
11. Too Fine >5mm, Too Coarse <5mm and High % Fracture
12. Too Fine >5mm and Too Fine <5mm
13. Too Fine >5mm, Too Fine <5mm and Low % Fracture
14. Too Fine >5mm, Too Fine <5mm, Low % Fracture and Presently Winter Season

After developing the belief network structure and incorporating probabilities into it, the next step was to test the network to evaluate the accuracy of its suggestions.

6.6 Testing the Belief Network

The first test of the belief network was to see if it could suggest improvements to a similar simulation model upon which the model was based. A model similar to the one used as a basis for developing the belief network was used for the first test. The researcher was familiar with the layout but not the characteristics of the plant components already modeled into the layout. Using the belief network, the researcher was able to make informed decisions and arrive at a product that met specifications after only six simulation iterations. After completing two more, the gradation was further optimized between the high and low specifications. For each simulation iteration, only one specific component change to the plant was made as it pertained to the suggestion from the belief network. It is possible to give evidence for more than one factor of the network, however, for simplicity's sake, only one result was evaluated during this test.

One must keep in mind that any suggestion from the network was repeated if necessary until further improvements to the gradation were no longer realized. Then, if possible, the network's next suggestion was implemented to improve the gradation results.

Table 6-14 Testing the Belief Network

Iteration No.	Gradation Results	First Possible Characteristic Change Suggested By Network
Plant As-Is	Too Coarse >5mm	Bottom Deck of Screen 2 is Too Large (changed size from 0.5 to 0.4375)
1	Too Coarse >5mm	Bottom Deck of Screen 3 is Too Large (changed size form 0.5 to 0.375)
2	Too Coarse >5mm	Screen 3 Bottom Deck Has Light Wire (changed from standard to coarse wire)
3	Too Coarse >5mm	Screen 3 Top Deck Has Light Wire (changed from standard to coarse wire)
4	Too Coarse	Too Little Sand

	<5mm	(increased flow of sand to final product from 50% to 70%)
5	Too Coarse <5mm	Too Little Sand (increased flow of sand to final product from 70% to 95%)
6	Within Specifications (decided to continue)	Top Deck of Screen 3 is Too Large (changed top deck from 0.5 to 0.375)
7	Better	Bottom Deck of Screen 3 is Too Large (changed bottom deck from 0.375 to 0.25)
8	Best	

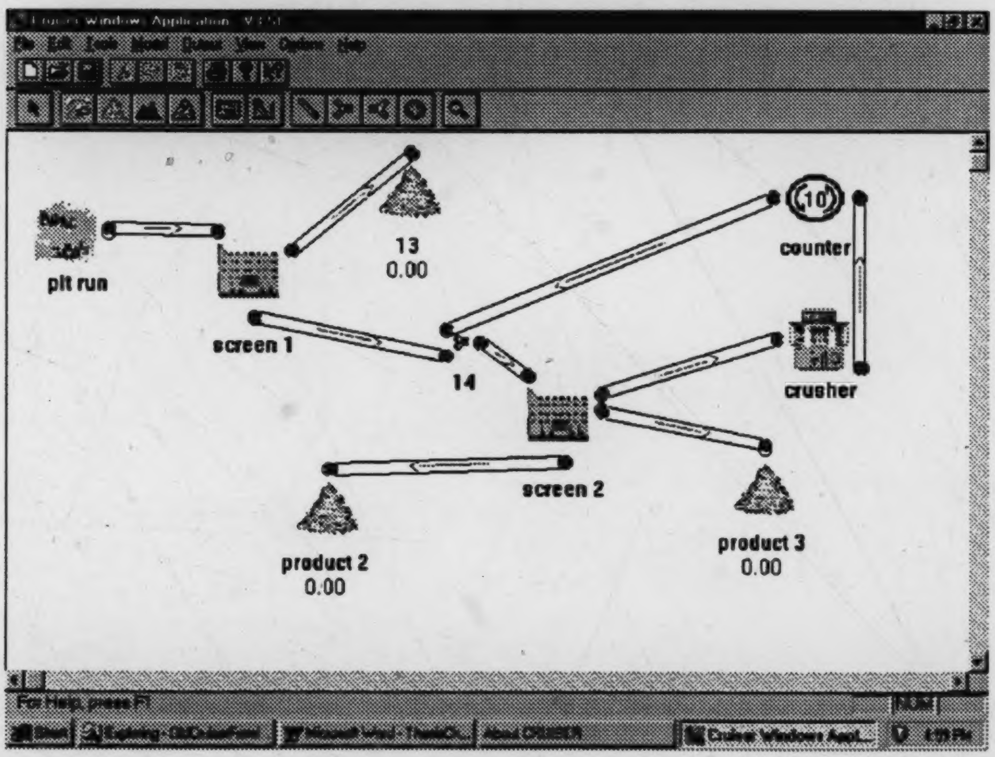

pit run
13
0.00
10
counter
screen 1
14
crusher
screen 2
product 2
0.00
product 3
0.00
For Help, press F1
NUM

Figure 6-4 – Initial Plant Layout

The second test of the belief network was to see if it could suggest improvements to a simulation model with fewer components than the one on which the model was based. The plant, its characteristics, and its specifications were used as provided in a graduate course. The plant layout of the model can be seen in Figure 6-4. Following the material flow through the system it is clear that product #3 is produced first, followed by product #2. The first thought was to view the product gradation of product #3 first, use the belief network, and make changes until it met specifications, and then repeat the procedure for product #2. Instead the researcher tried to implement the suggestions of the belief network, implicitly focussing on the pieces of equipment the network was referring to for the suggestions. After the first simulation run it was discovered that both product #2 and #3 were too fine in the >5mm range. The first suggestion made by the network was to increase the screen size on the intermediate screen (i.e., screen 2) on the second deck. For this plant layout there is no intermediate screen and so the suggestion does not apply and is skipped. The second suggestion was to increase the setting on the finishing crusher. This was done, making the product only minutely coarser. As a result the researcher moved onto the third suggestion, which was to increase the screen size on the finishing screen (i.e. screen 3) on the second deck. This suggestion was repeated for the next two simulation runs until product #3 met specifications and then continued with the same suggestion until product #2 also met specifications. Overall, the belief network functioned well, even for a layout producing two products at once. Using the belief network model the researcher arrived at a product that made specifications in four iterations. Without the model it took the researcher six iterations to solve this problem using trial and error. Another benefit of using the network was that a more optimal solution than the one obtained by the trial and error was gained. A gradation that met specifications could have been reached after only two simulation runs if the screen size was sufficiently increased. This shows that the number of simulation runs depends not only on the number of suggestions made by the belief network, but also on the number of iterations for which the user remains on the same suggestion. The user must use his/her own discretion as to how fast a suggestion is implemented so as to optimize the gradation with a given suggestion from the network. It is also up to the user to determine when to continue on to the next suggestion from the network.

6.7 Using the Belief Network

It is suggested that, when using the model, only one of the four primary nodes containing "out of specification" results should be pursued at any one time. A novice user could also have two combinations in the model. Only where both sides of the gradation (i.e., >5mm and <5mm) are either too coarse or too fine does the model handle similar causal factors. Any combination of too coarse and too fine together would suggest to the user a greater number of possible changes that could be made at one time. Here, the user would be making corrections to both ends of the gradation more efficiently. However, the user is less likely to understand the effects of each model suggestion. It is understood, however, that the user cannot suggest to the model that the gradation is both too coarse and too fine on the same side of the gradation, either the <5mm or >5mm side. A more experienced user might not have too much difficulty in making multiple changes at once while maintaining what effects the changes are making to the gradation. When assessing a given plant layout with specifications for more than one product, it is recommended that the user implement suggestions while observing the coarser product (i.e., product made first) until the specifications for this product are met. Then the user should implement changes to the finer product (i.e., product made second) until the specifications are also met. The user should monitor the effects of implementing suggestions on the coarser product, whose specifications are already met, while trying to meet the specifications of the finer product. For some plant layouts, implementing the suggestions as stated may bring the gradation for both products within the specifications at the same time.

It was decided not to implement an automated process using belief networks, mostly because there is more than one specific way to optimize the characteristics of a given plant. The user can use the belief network only as a guideline in the decision-making process and can implement a given suggestion as quickly as he/she desires. There are different combinations of plant characteristics that will produce the same product gradation, and there are a number of different layouts that require different suggestions to reach a product meeting specifications with a minimal number of iterations. Sometimes the user for either logistic reasons or out of user preference does not pursue a suggestion made by the network. As demonstrated in the testing phase, belief networks can assist a user of CRUISER in attaining the parameters of a plant required to produce a product within specifications. However, incorporating the belief network into the CRUISER program could allow for future automation of changes to the plant parameters. For automation purposes, the researcher suggests that suggestions to the model be implemented in small steps, since it would only involve computational time and not human resources. When a using the model a user may want to implement changes at a faster pace, which is likely to achieve the same result and much more efficiently.

Some of the heuristics rules for using the model were developed with the help of Bill Laisse. His input was needed because a causal node might have a greater probability of being the cause while another causal node may be adjusted to help the gradation meet specifications. This is done because in the field some adjustments to the plant are executed more easily and quickly than others. For example, if the product has too high a percentage of fractures, then the causes are either a lack of sand or the splits on the bottom deck of the finishing screen are too small. The field personnel would attempt to fix the problem by adjusting the percentage of sand entering the final product before stopping the plant and changing screens, although the probabilities would suggest otherwise. Most of the situations were accounted for within the model using probabilities and dividing up the causal nodes into the different scenarios of use. The user must still employ some logical thinking as to which parameters are likely to be changed at a plant to minimize plant downtime for example. In most cases, the user will not have the knowledge to make these logical decisions until some actual plant experience is gained. This is where the heuristic rules built within the model come into play.

However, to use the model developed in this research a set of steps should be followed. These steps are listed below. They describe how the model is to be used independantly from the CRUISER program and also how the model could be incorporated within the program. Before using the model, the user must set up an aggregate production model within CRUISER and run one simulation to get one set of gradation results. It is also required that the user has product specifications or has at least some indication of where he/she wants the gradation to be. Step #1 describes what the user would do if the belief network model was incorporated with CRUISER. Some information is given in step #4 if the model is not incorporated with the CRUISER program.

Step #1: The user must inform the model as to the layout of the model and what equipment components exist within the model. To use the developed belief network, the CRUISER model must not have more than three screens or two crushers. The network model within CRUISER would require this information to eliminate some of the possible suggestions from the network.

Step #2: The user must decide which of the three scenarios applies before implementing the belief network for suggestions to the model that he/she has developed. Depending on the scenario chosen, certain causal nodes may be ignored. These can either be made known to the user in the form of a user manual or may be hard coded into the CRUISER program. If the belief network model is incorporated into the CRUISER program, each causal node may be given a code that corresponds to each scenario. If the chosen scenario does not request a specific causal node, it will be left out of the suggestion list to the user when the model is activated. Referring to Table 6-11 through Table 6-13 it can be seen that the node are described as either characteristic or diagnostic in nature. This is denoted by a "C" or a "D" in the tables and can be used to describe all three possible scenarios for which the model is developed for.

Step #3: The user must then decide which evidence nodes he/she will provide evidence to. This can be anywhere from one to four nodes based on the developed network and the chosen scenario. A novice user would probably want to enter evidence into the network in only one node at a time. A more experienced user may want to enter evidence in two or more nodes. If there is more than one evidence node to pursue, a novice user should start with the evidence node that applies to his/her gradation according to the following list:

1. Too Coarse >5mm
2. Too Fine >5mm
3. Too Coarse <5mm
4. Too Fine <5mm

After continuing past this step the user must not change the scenario that he/she has chosen.

Step #4: The user must then evaluate the suggestions provided by the belief network and make one change at a time to the model. How great a change should be left up to the user's discretion. For automation purposes, the researcher suggests that suggestions to the model be implemented in small steps since only computational time would be spent, and not any human resources. However, when a user is using the model he/she may want to implement changes at a faster pace, which is very likely to achieve the same result and much more efficiently. A user may want to experiment with implementing more than one suggestion from the model. This may be acceptable for a more experienced user but is not recommended for a novice user. Implementing too many suggestions at once would lead to confusion instead of an increased understanding of what changes affect the gradation more significantly.

When any causal node is evaluated and is found to be either true or the likely cause, then all preceding nodes connected to the causal node must be evaluated before continuing onto the next most probable first level causal node. The second level causal nodes should only be evaluated if the first level causal nodes are being evaluated.

If the belief network model is not incorporated within CRUISER, the user would then have to ignore certain suggestions depending on the numbers of screens and crushers in the model the user is evaluating. Refer to Table 6-15 as a guide to ignoring suggestions from the network if there are less than three screens or less than two crushers in the plant layout.

Table 6-15 Suggestions for Different CRUISER Scenarios

Different Equipment Scenarios	Suggestions to Use
3 Screens	All Screen Suggestions
2 Screens	Only Screen 1 and 3 Suggestions
1 Screen	Only Screen3 Suggestions
2 Crushers	All Crusher Suggestions
1 Crusher	Only Crusher2 Suggestions

Step #5: The CRUISER simulation model must be run at this time to arrive at a change in gradation. This will allow the user to go back to Step #3 to use the belief network again. If the information in Step #3 does not need to be changed, the user may continue implementing a network suggestion to a greater degree or proceed on to the next suggestion by the network. Steps 3 through 5 may be repeated until the product meets specifications. The user must then decide to stop the optimization process or further optimize the gradation within the specifications. This can be done by progressing along the rank order of suggestions provided by the belief network.

6.8 Conclusion

The developed belief network has proven to be an effective diagnostic model for quickly attaining a final product that meets specifications. Some of the suggestions to the user of the CRUISER program are of characteristic in nature to represent an aggregate plant setup situation. Other suggestions are diagnostic in nature to represent problems encountered during actual plant operation, which may cause the final product to become unacceptable with respect to the specifications. It is recommended that further testing of the developed model be done with other aggregate plant configurations and with larger plants to assess its applicability. This future research could serve to expand or modify the developed model to increase its accuracy and applicability to other aggregate plant models. Visual Basic code could be written to incorporate the belief network within CRUISER to fully automate the equipment parameter selection for attaining a final product gradation that meets specifications. This will greatly enhance the power of the developed model to assess several possible factors at one time and present more than one possible final solution to the user. The researcher views this process as being an iterative one that is performed a specified number of times by the user. The program will monitor the equipment parameter chosen at a particular step in the optimization process and will not be allowed to repeat this selection for every iteration following unless a previous step selection between the two respective iterations was different. This would allow the user to select the combination of plant parameters that are most favourable with respect to the aggregate being processed and other economic related concerns in making the desired final product. For example, a thin wire screen might be suggested by the model as a final equipment parameter, but these screens may not be economical for use with the raw product being used (i.e., an abrasive material), so this final setup output by the simulation could be rejected by the user.

7.0 Final Discussion

7.1 Summarized Findings

The focus of this research has been to evaluate an aggregate production simulation program called CRUISER and develop models using artificial intelligence to improve it. A crushing model was developed to improve the accuracy and flexibility of the CRUISER program analyzed in this research. An optimization process was developed to add optimization and diagnostic features to CRUISER. The following describes the developments achieved and the findings reached during simulation analysis, and using neural and belief networks.

7.1.1 Simulation Analysis

CRUISER was found to be a user friendly and fairly accurate aggregate production simulation program. The output gradation is within ±10 % error over 40 to 60% of the sieve sizes on any given sample. The final product gradation predicts up to 16% coarser and 2% finer than the actual results over all samples used in this research. However, this range is still not accurate enough to predict a product gradation within specifications. CRUISER predicts the final product tonnage within an acceptable range. CRUISER does not predict the tonnage of intermediate streams very well, but predicts the final product tonnage with an average of 14%. An analysis of individual components revealed crushing errors to be more significant with the coarse crusher than with the fine crusher. Analysis of the screening process was difficult due to the high variability of raw product into the first screen. Complete analysis of the second and third screens could not be fully realized due to the inaccessibility of some product streams during the sampling process. However, in the portion of the screening analysis completed, gradation error was still significant. When evaluating the crushing process within CRUISER, modifications to the existing parameters resulted in slight gains in accuracy. When the screening process was evaluated, a discrepancy in screen efficiency was found. Allowing the user of the program to override the efficiency did result in an improvement in accuracy. However, this additional accuracy was not significant enough to predict a product gradation within specifications.

7.1.2 Neural Networks

On average, CRUISER predicts a coarser final product than what is actually produced. Additional modifications to the crushing and screening parameters in CRUISER did not significantly improve the gradation results of the program in this research. Focus was placed on the crushing process; neural networks were used to model this process. A prototype model was developed and the neural network obtained good results. A full-scale model was developed with greater variety in aggregate gradations. The neural network would not train with this data, and experimentation ensued. A smaller sample set was used to find that a single gradient model with 10 crusher settings yielded better error results than the prototype model. However, several single gradient models would be necessary to model one crusher type and setting. This was an impractical solution. The final model has comparable results as the single gradient model has fifteen gradient with only one crusher setting. Using this model would mean that ten neural networks would have to be developed for the crusher type, one for each setting. This final model can be used within the CRUISER program to enhance its crushing analysis capabilities.

7.1.3 Belief Networks

The developed belief network has proven to be an accurate diagnostic model that quickly attains a final product that meets specifications. Expert information on how changes in plant parameters affect the final product gradation was obtained and incorporated with the model in the form of nodes and probabilities. The network nodes were categorized into two types for two different types of users of the model. Some of the suggestions for the user of the CRUISER program are characteristic in nature to represent an aggregate plant setup situation. Other suggestions are diagnostic in nature to represent problems encountered during actual plant operation, which may cause the final product to become unacceptable with respect to the specifications. Incorporating the belief network within CRUISER and fully automating the optimization process will greatly enhance the power of the model. This will allow a user to assess several possible factors at one time and will present more than one possible final solution to the user. This would allow the user to select which combination of plant parameters that are most favourable with respect to the aggregate being processed. The selection could include other economic concerns related to making the desired final product. The use of belief networks to incorporate expert decision making into CRUISER has made the program more useful in terms of optimizing the equipment parameters for plant setup or during plant operation.

7.2 Contributions

A review of other aggregate production simulation models and a comparison with CRUISER was done to evaluate the accuracy and general modeling of existing programs. A neural network was developed as a basis to replace the crushing modeling process within CRUISER to increase accuracy and reflect actual crushing processes. The network can model the crushing process with additional plant specific information much more accurately than the traditional method of using aggregate production handbooks as a guideline for the gradation output results.

A belief network was developed and proven to be an accurate diagnostic model in quickly attaining a final product that meets specifications. The network nodes were categorized into two types with different users of the model in mind. The developed network handles three different user scenarios-educational, plant setup, and plant operation. Further testing of the developed model with other aggregate plant configurations and with larger plants will add to its ability to optimize other aggregate models. The use of the developed network incorporates expert decision-making into CRUISER and has added an optimization feature for use during plant setup or plant operation.

The main objective of this research was to assess the ability of CRUISER to model the aggregate production process, make analysis improvements, and add additional functions to further its applicability and usefulness for the aggregate industry. All of these objectives were achieved with this research and will contribute to the development of CRUISER as an accepted aggregate production modeling program within industry.

7.3 Final Comments

This research has shown the accuracy of the aggregate production simulation program, CRUISER, in actual use. Areas of improvement to the program were identified to increase the accuracy of CRUISER. This research has also proven the relevance and applicability of artificial intelligence for aggregate production modeling. Two primary forms used were neural networks and belief networks. The advantages of neural networks for modeling the crushing process more accurately will be reflected in the final product results. Neural networks will also serve to instigate a method of actual data collection by CRUISER users, which will be very beneficial, especially in the long term. Additional neural networks should be developed using the generated data process discussed in this research as an alternative to obtaining actual crushing gradation data to train networks. Additional neural networks for other crusher types and settings will need to be developed. The collection of actual data will allow for more accurate modeling but is not absolutely necessary. One option could be that neural networks would only be developed for common crusher types and settings. The remainder would have the neural network framework in place to incorporate a user's crushing data into the program through the use of a neural network trainer. A neural network trainer within the CRUISER program would allow a user to add additional data to an existing network, retrain the network, and use the developed crushing network. The user could select from three options when configuring a crusher within CRUISER. The traditional method of using gradation data from an aggregate crushing handbook would be one analysis option. A second option would be to select an already trained neural network for analysis. The third option would be to add additional data to an existing network and retrain it before use in the analysis. It is recommended that users develop their own data for the crusher types and settings they use most often. This would result in a neural network with better results than just adding a few sets of data onto an already developed network and retraining it. Obtaining data and developing a network from scratch will allow for the neural network to become more aggregate pit, equipment, and product specific. This will, in turn, create the most accurate and meaningful crushing analysis for the user of the CRUISER program. Finally, further work integrating the framework of the developed neural network into the CRUISER program will need to be done in order to take full advantage of this research.

The use of belief networks in optimizing aggregate plant setup or operation has been demonstrated in this research. The use of this model in combination with CRUISER will serve to automate the process of optimizing the equipment parameters required to achieve a final product gradation that meets specifications. Additional testing of the belief network model with other plant configurations and larger plants than the ones used in this research may be beneficial, further enhancing and expanding the existing model. This future research could serve to expand or modify the developed model to increase its accuracy and applicability to other aggregate plant models. The developed model can be easily modified to add findings from future research on different plant models. Further work automating the use of the developed belief network by integrating it within CRUISER will need to be done in order to take full advantage of this research. This will involve using Visual C++ and Visual Basic code to link the two models together in an efficient manner, minimizing the computational time required for the optimization process. An overview of how the two models may interact with one other is presented in this research. However, additional processes of how the model is used to present several possible equipment parameter scenarios should still be investigated.

This research has shown the usefulness and predictive power of artificial intelligence in aggregate production modeling. Artificial intelligence was used in conjunction with the CRUISER program in two different manners. One of these is a neural network, which was used to improve the crushing analysis within CRUISER. The second is a belief network, which was used to incorporate expert decision-making to develop a semi-automated process of attaining the right combination of plant parameters to achieve a final product that meets specifications. Through the use of both models, the aggregate industry will benefit from additional modeling accuracy and automation in obtaining the desired product results through the use of CRUISER.

Overall, the incorporation of these two forms of artificial intelligence into the CRUISER program will result in additional modeling accuracy, as well as creating a useful diagnostic and education tool. These improvements will encourage industry acceptance and increase the reliability of CRUISER and will serve to improve this simulation tool for the aggregate production industry.

8.0 References

AbouRizk, S.M., Hajjar, D., (1995) "CRUISER: A Crushing Simulation Environment Model", software developed by the Construction Engineering and Management Program of the Civil and Environmental Engineering Department, University of Alberta

AbouRizk, S.M., McCabe, B., Saadi, W., (1996) "Estimating Trenching Productivity Using Neural Networks", Third Congress on Computing in Civil Engineering, ASCE, pp 220-226

Al-Tabtabai, H., Kartam, N., Alex, A.P., (1996) "Neural Networks for the Identification and Control of Quantity Variance in Construction Projects", Third Congress on Computing in Civil Engineering, ASCE, pp 227-232

Canalog, E.M. and Geiger, G.H., (1973) "How to Optimize Crushing and Screening Through Computer-Aided Design", *Engineering and Mining Journal*, Vol. 174, pp 82-87

Cedarapids Inc., (1994) "Compucrush", Aggregate Production Software, Version 5.14

Cedarapids Pocket Reference Book, Fourteenth Edition, Cedarapids Inc., Cedar Rapids, Iowa U.S.A.

Charniak, E., (1991) "Bayesian Networks without Tears", *AI Magazine*, Vol. 12, No. 4, pp 50-63

Chehayeb, Nader, (1996) "Practical Analysis of Case Studies", I.C.E.S. Report on CRUISER

Choa, L.C., and Skibniewski, M.J., (1993) "Estimating Construction Productivity: Neural Network Based Approach", *Journal of Computing in Civil Engineering*, Vol. 8, No. 2, pp 234-251

Chong, H.G., and Walley W. J., (1996) "Rule-based versus probabilistic approaches to the diagnosis of faults in wastewater treatment processes", *Artificial Intelligence In Engineering*, Vol. 10, No. 3, pp 265-273

Du Plessis, I., (1994) "Optimizing Crushing Plant Performance using a Dynamic Simulation", Master of Engineering Thesis, Faculty of Engineering, University of Pretoria

Duursma, G.R., (1990) "Crushing and Screening Models For Simulation", Masters of Engineering Thesis, Faculty of Engineering, University of Pretoria

Epstein, B., (1948) "Loagrithmico-normal Distributions in Breakage of Solids", *Industrial Engineering and Chemistry*, Vol. 40 pp 2289-2291

Flood. I., Kartam, N., (1994) "Neural Networks in Civil Engineering II: Systems and Application", *Journal of Computing in Civil Engineering*, Vol. 8, No. 2, pp 149-162

Garret, J.H., (1992) "Neural Networks and their Applicability within Civil Engineering", *Computing in Civil Engineering*, Proceedings of Eighth Conference, Dallas, Texas, pp 1155-1162

Gurun, T., (1973) "Design of Crushing Plant Flowsheets by Simulation", Application of Computer Methods in the Mineral Industry, pp 91-98

Hajjar, D., and AbouRizk, S.M., (1997) "Modeling and Analysis of Aggregate Production Operations", Paper Under Review.

Hancher, D.E., and Havers, J.A., (1972) "Mathematical Model of Aggregate Plant Production, ASCE, New York, United States

Heckerman, D., Wellman, M.P. (1995) "Bayesian Networks", Communications of the ACM, Vol. 38, No.3, pp 27-30

Heikkila, P., (1992) "Improving the Quality of Crushed Rock Aggregate", Ph.D. Thesis, Finnish Academy of Technology, Helsinki, Finland

Henrion, M., Breese, J.S. Horivitz, E.J., (1991) "Decision Analysis and Expert Systems", *AI Magazine*, Vol. 12, No. 4, pp 64-91

Karra, V.K., (1979) "Development of a Model for Predicting the Screening Performance of a Vibrating Screen", CIM Bulletin, pp 167-171

Lynch, A.J., (1977) "Mineral Crushing and Grinding Circuits: Their Simulation, Optimisation, Design and Control", Elsevier North-Holland, New York, NY

McCabe B., (1997) "An Automated Modeling Approach for Construction Performance Improvement Using Computer Simulation and Belief Networks", Ph.D. Thesis, University of Alberta

Microsoft Corporation, (1996) "MSBN – Microsoft Bayes Networks", Bayesian Belief Network Construction and Evaluation Software, Version 1.001

Moselhi, O., Hegazy, T., and Fazio, P., (1990) "Neural Networks as Tools in Construction", *Journal of Construction Engineering and Management*, Vol. 117, No. 4. pp 606-623

Murtaza, M.B., Fisher, D.J., (1993) "Neuromodex - Neural Network System for Modular construction Decision Making", *Journal of Computing in Civil Engineering*, Vol. 8, No. 2, pp 221-233

Pioneer Pocket Reference Book, Twentieth Edition, Portec/Pioneer Division, Yankton, South Dakota U.S.A

Poole, David L., Mackworth, Alan, Goebel, Randolph G., (1998) "Computational Intelligence: A Logical Introduction, Oxford University Press, New York

Portas, J.B., (1996) "Estimating Concrete Formwork Productivity", M.Sc. Thesis, University of Alberta

Savin, D., Alkass, S., Fazio, P., (1996) "Construction Resource Leveling Using Neural Networks", *Canadian Journal of Civil Engineering*, Vol. 23, pp 917-925

Sawhney, A., AbouRizk, S.M., Dozzi, S.P., (1993) "Forecasting Construction Cost Escalation", University of Alberta Unpublished Internal Report, Construction Engineering and Management

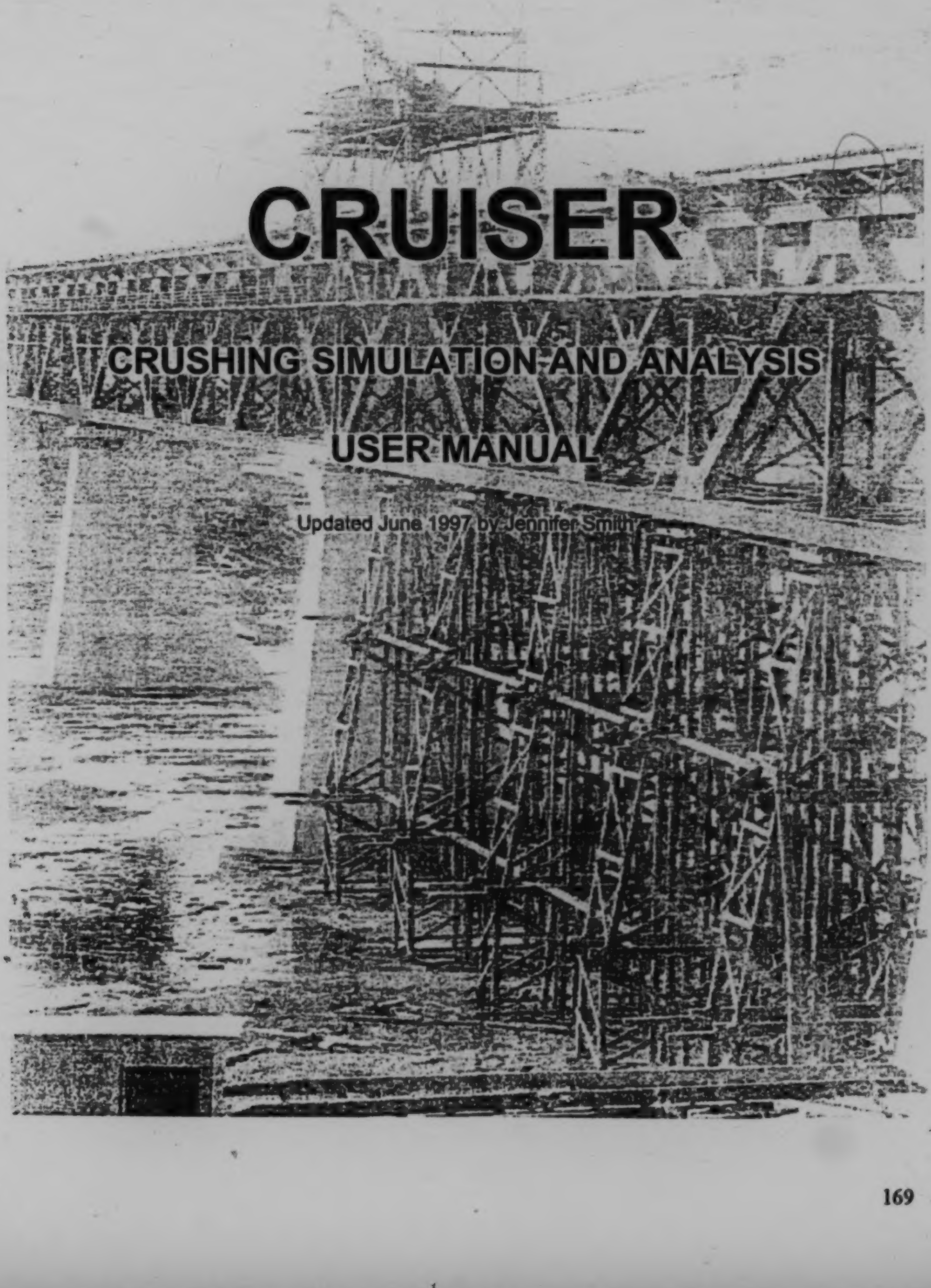
Spiegelhalter, D. et al., (1989) "Assessment Criticism and Improvement of Imprecise Subjective Probabilities for a Medical Expert System", In Proceedings of the Fifth Workshop on Uncertainty in Artificial Intelligence, 335-342. Mountain View, California: Association for Uncertainty in Artificial Intelligence.

Wales, R.J., (1993) "Modeling Construction Productivity for Computer Simulation", M.Eng. Thesis, University of Alberta

Whitby, K.T., (1958) "The Mechanics of Fine Sieving: Symposium of Particle Size Measurement- A.S.T.M. Special Technical Publication, Vol. 234, pp 3-24

APPENDIX A

CEM



CRUISER

CRUSHING SIMULATION AND ANALYSIS

USER MANUAL

Updated June 1997 by Jennifer Smith

CEM

CRUISER

CRUSHING SIMULATION AND ANALYSIS

USER MANUAL

Updated June 1997

Table of Contents

Table of Contents ii

Introduction 1

Installation 1
System Requirements *1*

Installation Procedure *1*

Technical Support *1*

Getting Started 2
CRUISER Utilities: *2*

CRUISER Objects: *2*

The CRUISER Menu *3*

Creating Components *4*

Deleting Components *4*

Directing Stream Flow *4*

Analysis Process 5
CRUISER Components *5*

Conveyers 6
Raw Pile 8
Crushers 10
Screens 11
Product Piles 12
Surge Piles 13
Waste Piles 13

Simulation 14
The Integrity Checker *14*

Simulation *14*

Appendix A 15
Itype setting for single impeller impact crushers *15*

Itype setting for double impeller impact crushers *15*

Appendix B 16
Calibration of the Screens to Actual Plant Operations *16*

Appendix C 17
Transferring Information to and from Excel Spreadsheets *17*

Appendix D 18
Sample Screens from the Options Menu *18*

Introduction

CRUISER (Crushing Simulation Environment) is a Windows application designed to simulate the operations of a crushing plant. The objective of CRUISER is to provide managers with a tool to experiment with several possible alternatives for plant design. This program allows the simulation of many different situations to be performed very easily by changing the desired parameters and re-simulating. The application takes care of all the tedious and time-consuming work of analysis that would otherwise be done by hand.

The analysis of crushers and screens is based on empirical data collected from field operations.

Installation

System Requirements

- Pentium or better IBM compatible computer.
- A minimum of 16 MB of RAM (32 MB is recommended.)
- A hard disk with at least 15MB of free space available.
- SVGA or better resolution monitor.
- Microsoft Windows version 95 or higher.

Installation Procedure

- Run setup.exe from the first disk of the installation diskettes and follow the instructions.
- The setup program will copy all files to the specified directory and install various libraries in the Windows directory.
- Run cruiser.exe from the installation directory or use the newly created program shortcut.
- CRUISER will perform one time initialization the first time it is run.

Technical Support

If you have any comments or require any help with regards to any aspect of this program, please contact Dany Hajjar at:

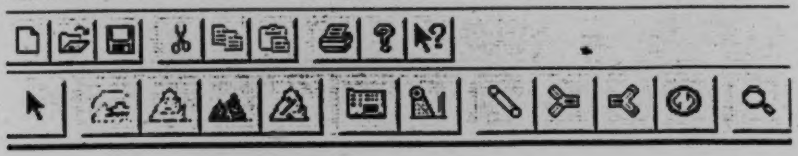
Email (Preferred)
hajjar@cem.civil.ualberta.ca

Mail
220 Civil/Electrical Building
University of Alberta
Edmonton, Alberta
Canada T6G-2G7

Phone
(403) 492-2276

Getting Started

When CRUISER is started a screen appears which contains the following toolbar.



Each button on the toolbar represents an object or a function in the CRUISER program. A list of the buttons and their functions follows:

CRUISER Utilities:

Zoom Lens (Right Click on work area to zoom in on screen objects, left click to zoom out.)

This button is used to return the mouse pointer to the default mode from any other mode. To do so, click on the button when the pointer is in an auxiliary mode.

Help Pointer. (To get help on a certain object, click this button then click again on the object with which you need help.

General information click this button and a box containing general information on the program will appear on the screen.

CRUISER Components:

Raw Pile

Product Pile

Waste Pile

Surge Pile

Crusher object

Screen

Conveyor

Add joint

Split Joint

The CRUISER Menu

The menu bar in CRUISER consists of the following categories and sub categories. (Explanation of the function of these sub categories will be given when necessary.)

- File
    - **New** File
    - **Open** Existing file
    - **Save** File
    - **Save** (File) **as** (the desired name here).
    - **Print**
    - **Print Preview**
    - **Print Setup**

- Edit
    - **Cut**
    - **Copy**
    - **Paste**
    - **Properties** – To edit the properties of an object, select that object, then select "Properties" from the "Edit" menu and the screen that allows for editing of properties will appear. For instance, to edit the properties of a raw pile object, select the raw pile, then select "Properties" from the "Edit" menu and the window shown in Figure 4 will appear.

- Tools – This menu performs the same function as the CRUISER object buttons that appear on the previous page. To place an object on the working area, go to the Tools menu and select the desired object then move the pointer to the place on the desktop where you wish to put the object, click the left mouse button and that object will appear.
    - **Screen**
    - **Crusher**
    - **Product Pile**
    - **Waste Pile**
    - **Raw Pile**
    - **Surge Pile**
    - **Conveyor**
    - **Add Joint**
    - **Split Joint**

- Model
    - **Check** (model) **Integrity** – This function is used to ensure that none of the necessary elements in the model are missing or not properly connected.
    - **Analyze** (model) **Streams** – This function initiates analysis of the model.

- Output
    - **Print**
    - **Print Preview**
    - **Enable All Nodes** – Select "Enable All Nodes" from the "Output" menu to have the "Display Results" box automatically checked for every component in the simulation.
    - **Disable All Nodes**– Select "Disable All Nodes" from the "Output" menu to have the check mark removed automatically from the "Display Results" boxes for every component in the simulation.
    - **Show** – To view the results of the most recent simulation, select Show" from the "Output" menu and the results box will appear.
    - **Hide** - To hide the results of the most recent simulation select "Hide" from the "Output" menu and the results box will disappear.
    - **Clear** – To clear the results from the Results box, select "Clear" from the "Output" menu.
    - **Settings** – Background information: When any conveyor is clicked with the right mouse button, the data for that conveyor is dumped into the results box and can be viewed. When "Settings" is selected from the "Output" menu, a window appears in which the option to have the Results box cleared before each dump. To have the results cleared from the box, ensure that a check appears in the "Clear Window Before Conveyor Dump" box.

- View – This menu allows for modification of the appearance of the program on the screen. It allows for zooming in and out as well as regulation of the appearance/disappearance of the toolbar and status bar.

    - **130 (%)**
    - **110 (%)**
    - **100 (%)**
    - **90 (%)**
    - **80 (%)**
    - **70 (%)**
    - **60 (%)**
    - **Toolbar**
    - **Status Bar**

- Options
    - **Sieves** – By selecting "Sieves" from the "Options" menu, it is possible to modify sieve sizes within the model. (See Appendix D for the screen that appears when "Sieves" is selected.)
    - **Desired** – By selecting "Desired" from the "Options" menu, it is possible to set the desired percentage pass low and the desired percentage pass high values for each sieve size. (See Appendix D for the screen that appears when "Desired" is selected.)

- Help
    - **Index**
    - **Using Help**
    - **About CRUISER**

Creating Components

- To place a new object in the working area, click on the corresponding button with the pointer. (After selecting the desired component, the cursor shape will change to reflect the current state.)
- Move the pointer to the place where you wish the object to be.
- Click the left mouse button and the object will appear on the screen. This newly created component will have default specifications. (To change the specifications, double click on the component (or selects "Properties" from the "Edit" menu). A dialog box specific to the type of object selected will be displayed. The specifications of that object can be modified in the dialog box.)
- Once placed, moving objects around the site layout can be accomplished using simple click and drag operations.

Deleting Components

- To delete a component, select the desired object by clicking on it, then press the "DEL" key or select "Cut" from the "Edit" menu.

Note: All information on connections to and from the deleted object will be deleted along with that object.

Directing Stream Flow

In CRUISER, conveyers provide the means of identifying the propagation and direction of streams. Each component posses one or more "connection circles". These are special sub-components that allow each object to be connected to other components through conveyers. To connect two components, create a conveyer and place its "source circle" on top of the "destination circle" of the source component and vice versa for the other side of the conveyer. An arrow from the source circle to the destination circle indicates the direction of the conveyer. When a connection between a conveyer and a component is established, the interior color of the connecting circle is changed to red. Any subsequent movement of the conveyer or the connected component can cause the connection to be deleted. To prevent disconnection, press the "Ctrl" key before selecting the object and hold it down while dragging the object.

Note: The crushing model can be as large as needed. (CRUISER supports scrolling and zooming capabilities.)

Analysis Process

The steps used to analyze a proposed plant are illustrated in Figure 1.

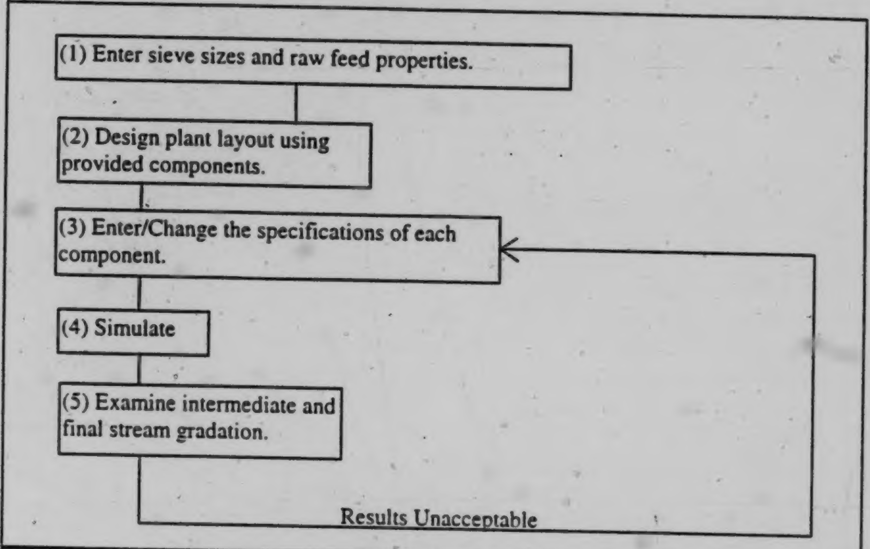

(1) Enter sieve sizes and raw feed properties.
(2) Design plant layout using provided components.
(3) Enter/Change the specifications of each component.
(4) Simulate
(5) Examine intermediate and final stream gradation.
Results Unacceptable

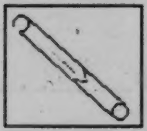
Figure 1 – Steps Used to Analyze a Proposed Plant

CRUISER Components

CRUISER provides a wide range of components that are used to represent an actual crushing operation. Each component has graphical properties used for display purposes and specification data that defines how it behaves during analysis. All components have a "name". This name is used as a title or description on the screen and in the analysis results. All the components also have the "Display Result" property that determines if the output specific to the component will be displayed or sent to the results file (if you have chosen to create one).

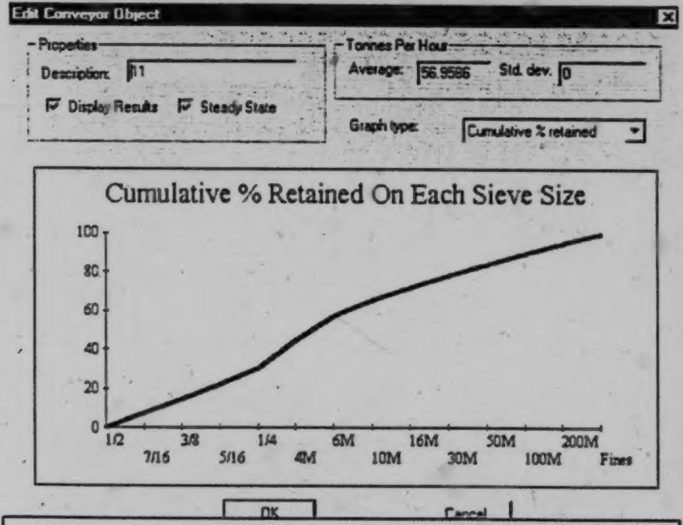
A warning message box may appear on the screen during simulation if the program detects the possibility that an error has occurred. You may continue on to view the simulation results after the warning message appears, or you may decide to go back and alter the parameters and run the simulation again. The following is an overview of the major components along with any properties that require clarifications:

Conveyers



Conveyers are used to define the propagation of stream flow throughout the plant. They can be used to connect components together or to connect components to piles. By moving one of the two end circles, the conveyer can be sized to whatever length (graphical) and angle is appropriate. Clicking and dragging the "body" of the conveyer will move the whole conveyer without changing the shape. Connecting two components is accomplished by placing one of the end circles on top of the destination circle of the destination component. (When a connection is present, the circle will be red. When there is no connection, the circle will be white.)

Conveyers also store the properties of the intermediate streams of the plant. After simulation, double clicking on any conveyer brings up the property dialog box as shown below in Figure 2 where stream gradation can be examined graphically.



Edit Conveyor Object

Properties

Description: 11

☑ Display Results ☑ Steady State

Tonnes Per Hour

Average: 56.9586 Std. dev. 0

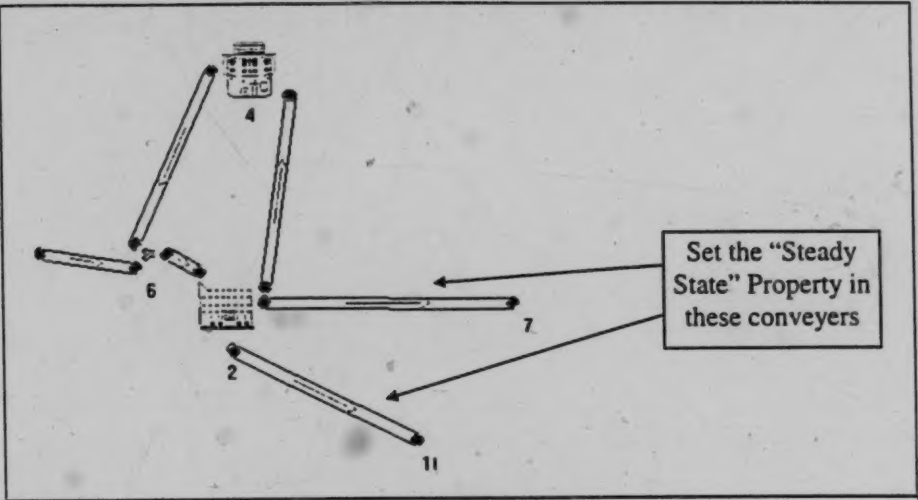
Graph type: Cumulative % retained

Figure 2 – Graphical Representation of Stream Properties

The Graphical display can be represented as a "Cumulative Percent Retained," or as a "Cumulative Percent Passing" graph. To select the type of graph displayed, use the "Graph Type" drop-down box.

Note: It is possible to view the average specific information on the selected stream. Click with the right mouse button on the desired conveyor and the following information will be displayed in a result box: actual tonnages retained, percent retained and cumulative percent retained on each sieve size.

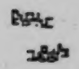
Whenever a conveyer represents the exit point of a cycle, it should be set as "Steady State" by ensuring that there is a check mark in the "Steady State" box (found in the window shown in Figure 2). Otherwise the closed circuit operation will not be analyzed correctly. All other conveyers should have the "Steady State" property unchecked. The diagram in Figure below demonstrates which conveyers should have the "Steady State" box checked.



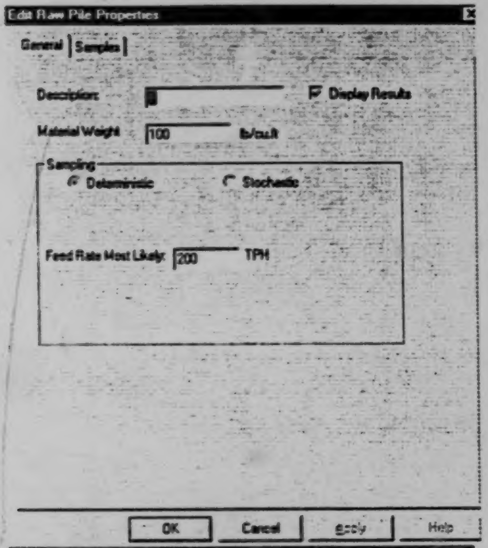
Set the "Steady State" Property in these conveyers

Figure 3 - Setting the "Steady State" Property

Raw Pile



Every project must have one and only one Raw Pile object. These are used to store the properties of the initial feed stream and represent the starting point of the simulation. When the "Raw Pile" object is double clicked, the screen shown in Figure 4 appears.



Description: 0

☑ Display Results

Material Weight 100 lb/cu.ft

Sampling

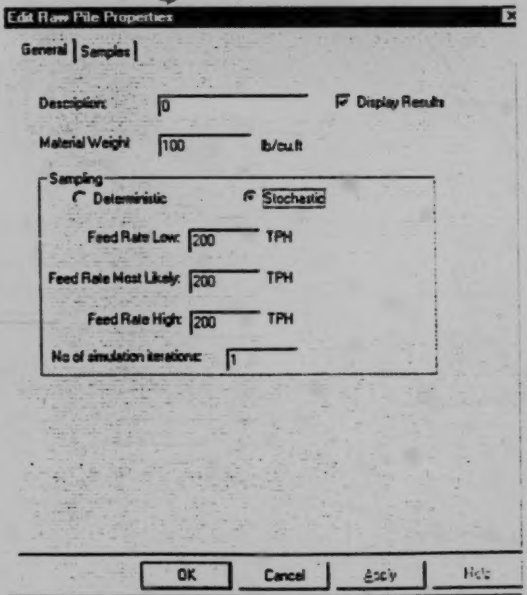
● Deterministic ○ Stochastic

Feed Rate Most Likely: 200 TPH

Figure 4 - The Deterministic Sampling Technique

CRUISER can be set to allow for the representation of uncertainty in input data. This can be accomplished by specifying a stochastic instead of a deterministic sampling technique. When stochastic sampling is selected (as in Figure 5), the low, high and most likely feed rates must be set along with the number of iterations. Once these are set, CRUISER will execute the simulation for a specified number of runs. To obtain the values for the above fields, a random sample of the input feed rate could be taken by obtaining 10 readings over the course of a day. For instance, the readings could be as follows: 200, 400, 450, 325, 350, 375, 425, 275, 350 TPH. The low feed rate would be 200, the most likely feed rate would be 347.5 (the average of all the samples) and the high feed rate would be 450.

There are two possible sampling techniques employed by CRUISER: deterministic and stochastic. The deterministic technique is used to analyze one sample. The stochastic technique is used to analyze more than one input. The stochastic technique gives an output with some variants from the most likely result. To set the sampling technique, double click on the source object and the box shown in Figure 4 will appear. Select the desired sampling technique from the screen shown in Figure 4. To have the program use the deterministic sampling technique, ensure that a dot appears in the "Deterministic" box (as in Figure 4). On this screen, also set the material weight and the most likely feed rate. When all information has been entered, select **"OK."**



Description: 0

☑ Display Results

Material Weight 100 lb/cu.ft

Sampling

○ Deterministic ● Stochastic

Feed Rate Low: 200 TPH

Feed Rate Most Likely: 200 TPH

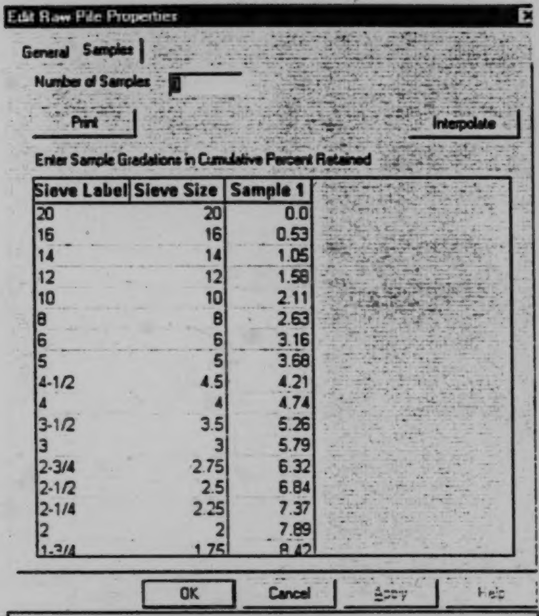
Feed Rate High: 200 TPH

No of simulation iterations: 1

Figure 5 - The Stochastic Sampling Technique

The production results from the stochastic sampling technique will be presented in the form of an average and a standard deviation. The gradation is presented as a 95% confidence interval and can be examined in the product pile.

To enter information on samples, select the "**Samples**" tab and the box shown in Figure 6 will appear on the screen.



Enter Sample Gradations in Cumulative Percent Retained

Sieve Label	Sieve Size	Sample 1
20	20	0.0
16	16	0.53
14	14	1.05
12	12	1.58
10	10	2.11
8	8	2.63
6	6	3.16
5	5	3.68
4-1/2	4.5	4.21
4	4	4.74
3-1/2	3.5	5.26
3	3	5.79
2-3/4	2.75	6.32
2-1/2	2.5	6.84
2-1/4	2.25	7.37
2	2	7.89
1-3/4	1.75	8.42

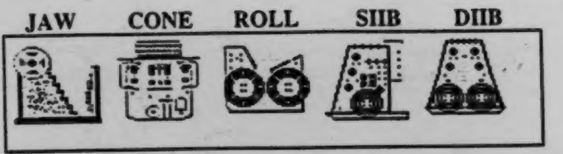
Figure 6 - The "Samples" Tab

Enter the data on the samples in the "Sample 1" column. If you do not have data for all of the intermediate sieve sizes, enter the data you do have and, providing that data for a range of sieve sizes is present, select the "**Interpolate**" button and the program will interpolate the data for all sieve sizes without data.

Interpolation can be executed in two ways, by a linear or duplication method. The linear method interpolates data between the entered data points in a linear fashion. The duplication method repeats the manually entered information from the previous cell until a cell with manually entered information is encountered. When all the data has been entered, select the "**OK**" button. To print the chart, select the "**Print**" button.

See Appendix C for information on transferring information to and from Excel spreadsheets.

Crushers



There are four types of crushers (shown above) that can be simulated by CRUISER. When a crusher object is double clicked, the window shown in Figure 7 appears. Information falling under the following categories can be entered from this screen. (Not all of the categories listed below will apply to every crusher. The categories that apply to the specified crusher will appear on the screen when that crusher is selected from the drop-down list in the "Type" box.)

Type: The type of crusher being used (Jaw, Cone, Roll Single Impeller Impact Breaker (SIIB), or Double Impeller Impact Breaker (DIIB)).

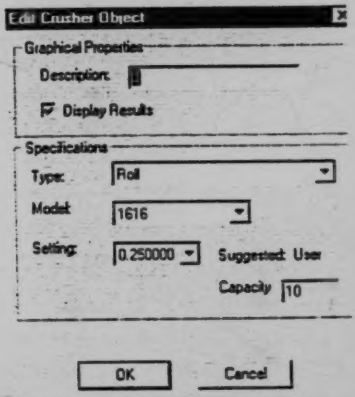
Model: The model number of the crusher being used in the simulation. (This section only applies to Roll, SIIB and DIIB.)

Size: The size of the crusher. (This only applies to the Cone Crusher.)

Dimensions: The dimensions of the crusher. (This only applies to the Jaw crusher.)

Setting:
The crusher setting in inches.

Capacity:
The estimated capacity of the crusher in TONNES per hour. The suggested value is placed above this entry box. The value is extracted from the Cederapids Pocket Reference books and is based on the dimensions and settings of the crusher.


Edit Crusher Object

Graphical Properties

Description:

☑ Display Results

Specifications

Type: Roll

Model: 1616

Setting: 0.250000

Suggested: User

Capacity 10

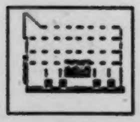
Figure 7 - The Edit Crusher Object Window

Note: This value does not have to be used; it's only a recommendation.

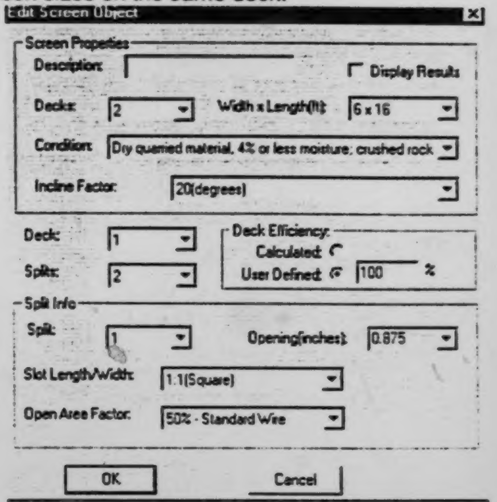
Itype:
This is the operating speed combination for single (1 - 8) and double impeller impact breaker (1 - 11). Each type corresponds to a specific relationship of input-output that is used to predict the product gradation. (See tables in Appendix A.) (This category only applies to the SIIB and the DIIB crushers.)

Note: For information on customizing crusher performance specifications by replacing gradation data, see Appendix B.

Screens



Screen objects in CRUISER have the ability to model screens with multiple decks and splits. Multiple splits allow for the representation of varying mesh sizes on the same deck.



Description:
☐ Display Results
Decks: 2
Width x Length(ft): 6 x 16
Condition: Dry quarried material, 4% or less moisture; crushed rock
Incline Factor: 20(degrees)
Deck: 1
Splits: 2
Calculated: ○
User Defined: ● 100 %
Split: 1
Opening(inches): 0.875
Slot Length/Width: 1:1(Square)
Open Area Factor: 50% - Standard Wire

Figure 8 - The Edit Screen Object Window

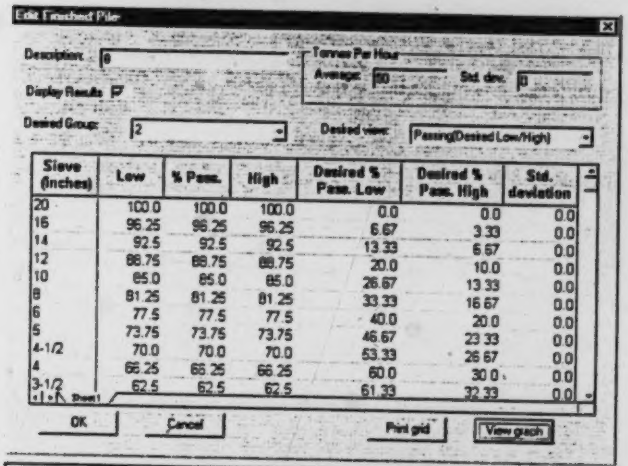
When the Screens object (shown above) is double clicked, the box shown in Figure 8 appears on the screen. The following information can be entered in this box:

- A description of the screen. If no description is entered, the description will remain as the number it was assigned upon placement.
- In the "Decks" box, set the number of decks (up to four decks maximum).
- The condition of the material in the "Condition" box.
- The incline factor of the conveyor in the "Incline Factor" box. This box indicates whether the screen is of a horizontal type or is inclined at some specified angle.
- In the "Deck" box, from the drop down list, select the deck for which you wish to enter information. The information shown in the window will represent the properties of the active deck. The active deck can be changed using the "Deck" drop-down list.
- In the "Splits" box, enter the number of splits to be present in the specified deck (up to a maximum of four splits per deck).
- In the "Split" box, select the split for which you wish to enter information. The information shown in the window will represent the properties of the active split. The active split can be changed using the "Split" drop-down list.
- Enter the desired information about the specified split in the "Split Info" section.
- If all the splits on the specified deck have the same properties you do not need have more than one split for that deck. If you have multiple splits with different properties, select the next split for which you wish to enter information and enter that information (If you neglect to set different properties for each split they will all have the same properties as the first split by default.)
- When all information on the splits on the specified deck has been entered, select the next deck for which you wish to enter information. If all the decks on the specified screen have the same properties you do not need the set the properties for the other decks.
- When all information has been entered on the decks and their splits, select the "**OK**" button.

Note: For information on calibrating screen efficiency within the model, see Appendix B.

Product Piles

Product piles provide detailed information about the stream properties of the final product. After the simulation is complete, the dialog box displays the stream gradation and tonnage in a table or in graphical format (shown in Figure 10). The output can be viewed alongside a desired group or envelope. By selecting a group from the "Desired Group" combo box, the information appears in the fifth and sixth columns.



Description: 8

Tonnes Per Hour

Average: 50 Std. dev. 0

Display Results ☑

Desired Group: 2

Desired view: Passing(Desired Low/High)

Sieve (inches)	Low	% Pass.	High	Desired % Pass. Low	Desired % Pass. High	Std. deviation
20	100.0	100.0	100.0	0.0	0.0	0.0
16	96.25	96.25	96.25	6.67	3.33	0.0
14	92.5	92.5	92.5	13.33	6.67	0.0
12	88.75	88.75	88.75	20.0	10.0	0.0
10	85.0	85.0	85.0	26.67	13.33	0.0
8	81.25	81.25	81.25	33.33	16.67	0.0
6	77.5	77.5	77.5	40.0	20.0	0.0
5	73.75	73.75	73.75	46.67	23.33	0.0
4-1/2	70.0	70.0	70.0	53.33	26.67	0.0
4	66.25	66.25	66.25	60.0	30.0	0.0
3-1/2	62.5	62.5	62.5	61.33	32.33	0.0

Figure 9 - The "Edit Finished Pile" Window

To print the chart shown in Figure 9, select the **"Print Grid"** button.

To edit the information in the "Desired % Pass High" and "Desired % Pass Low" use the "Options" menu. (For more information on the "Options" menu, see page thee.)

By pressing the **"View Graph"** button, a graphical representation of the gradation and the high and low specifications is presented (as shown in Figure 10).

To Print the Graph shown in Figure 10, select the **"Print"** button.

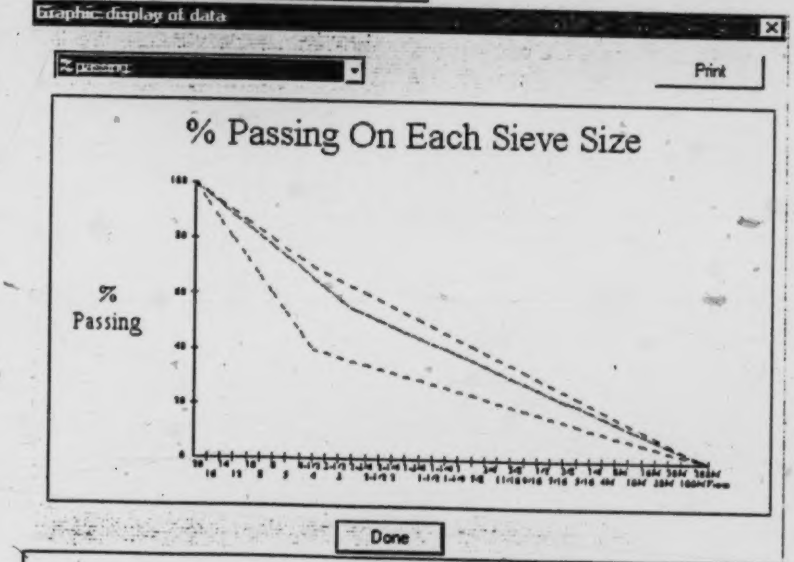
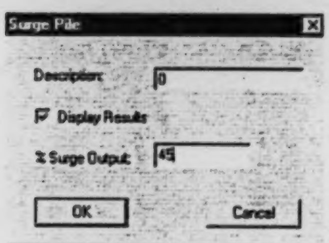


Figure 10 – Graphic Display of Product Pile Data.

Surge Piles

Surge Piles are inserted into the plant to control material flow at a specific location. The serge pile allows only a specified percentage of the input to pass along to the output. To set this percentage, double click on the serge pile and the box shown in Figure 11 (below) will appear.



Description: 0

☑ Display Results

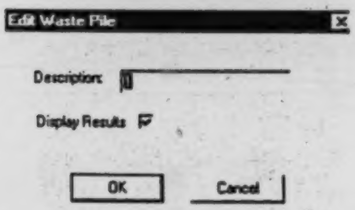
% Surge Output: 45

Figure 11 - Surge Pile Window

Enter the desired percentage in the "% Surge Output" box and select "**OK.**"

Waste Piles

Waste Piles are inserted into the layout of plants to collect unwanted products. They do not have properties. To view gradation data on the material entering the waste pile, double click on the conveyor that is connected to that waste pile. If you require more detailed information on the contents of waste piles, use a product pile instead and label it as a waste pile. This will enable you to view information on the material in the pile.



Description: 0

Display Results ☑

Figure 12 - The "Edit Waste Pile" Window

The only modification that can be made to the waste pile is the description and whether the results are displayed or not. When the waste pile is double clicked, the box shown in Figure 12 appears.

Simulation

The Integrity Checker

An integrity check can be performed on the model by selecting "Check Integrity" from the "Model" menu. This will display all the connection errors that are found in the model if any. The following is a list of errors that can occur along with explanation wherever necessary.

- "Unable to find any Raw Feed Piles."
  - Every model must have a raw input pile that defines the gradation, material weight, and tonnage of the initial feed stream.
- "Only one Raw Feed Pile allowed."
  - CRUISER currently supports only one raw feed per project.
- "Invalid Output Connection for Component <component name>"
  - The component <component name> was not connected appropriately. An error message specific to the component in question will be displayed following the above message, which details where exactly the problem is.

Simulation

CRUISER has the ability to analyze both open and closed circuit models. Open-circuit models involve straightforward analysis where each component is analyzed once. Closed-circuit models however involve special modeling considerations as discussed in the conveyer section of this user's guide. Closed-circuit models involve cycles where some of the crushed material is recycled back to a previous stage of analysis.

Once the plant design is complete, the analysis can be initiated by selecting "Analyze Streams" from the "Model" menu. This will start the integrity checker to make sure there are no errors before performing the analysis. (To run the integrity checker without running the whole analysis, select "Check Integrity" from the "Model" menu.) After the analysis is done the output window displays all the results from the components that had the "Display Results" box checked. At this point the stream gradation can be displayed at any conveyer by clicking on the desired conveyer with the right mouse button. This will dump the stored results to the output window. The gradation can also be viewed graphically by double clicking on the conveyer with the left mouse button. Final product results can be examined through the Product Pile object.

Appendix A

Itype setting for single impeller impact crushers

Model									Model
3020	700	600	500	400					3020
3026	700	600	500	400					3026
3623	630	590	510	470	430	390			3623
3633	630	590	510	470	430	390			3633
4325		650	600	550	500	450	400		4325
4336		650	600	550	500	450	400		4336
4326		650	600	550	500	450	400		4326
4340		650	600	550	500	450	400		4340
5348					440	410	380	350	5348
ITYPE	1	2	3	4	5	6	7	8	ITYPE

Itype setting for double impeller impact crushers

Model												Model
2222	700	650	600	550	500	450	400					2222
3042	700	650	600	550	500	450	400					3042
3645S												3645S
3645H		610	580	550	520	490	460	430	400	370		3645H
4350S		610	580	550	520	490	460	430	400	370		4350S
4350H					530	505	480	455	430	405	380	4350H
5060H					530	505	480	455	430	405	380	5060H
ITYPE	1	2	3	4	5	6	7	8	9	10	11	ITYPE

Appendix B

Calibration of the Screens to Actual Plant Operations

When selecting a user defined efficiency for a chosen deck within a screen object, the best source of information would be through actual sampling from the screen being modeled. This can be done by sampling the oversize stream of a given deck and knowing what screen sizes were used on this deck. From the weight of the sample obtained it can be determined what percentage of the entire gradation sample was in the oversize stream that actually should have been in the undersize stream based on the screen sizes. This percentage of material is considered to be the inefficiency of the deck being sampled. Therefore the efficiency of this deck is 100% minus the percentage of undersize in the oversize stream. This efficiency value can then be relied upon as the actual efficiency and can be entered into the model to enhance the accuracy of the analysis. Repeating the above process.

Entering Additional Actual Crushing Gradations into CRUISER

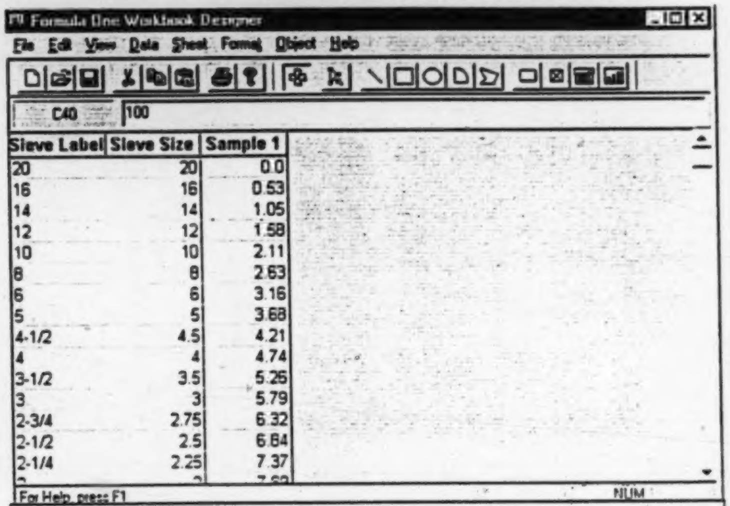
It might be found that in some situations the accuracy of the final product gradation contains some deviations from the actual observed results. Some of this error can be attributed to the gradations the program uses based on certain types of crushers and their crusher setting. If desired, more specific crushing data for a given crusher and setting can be determined in the field and added to the program. To enter additional or replace existing gradation data for crushers the following crusher specific data files can be edited by opening them using any given text editor (such as Notepad).

Type of Crusher	Data File to Edit
Jaw Crusher	jawi.dat
Cone Crusher	conei.dat
Roll Crusher	rolli.dat
Single Impeller Impact Breaker	sinpacti.dat
Double Impeller Impact Breaker	doupacti.dat

Appendix C

Transferring Information to and from Excel Spreadsheets

To transfer information to and from an Excel spreadsheet, Click on the "Sample 1" column in Figure 13, then, double click on that column with the right mouse button and the "Workbook Designer" box (shown in Figure 13) will appear. From this screen, highlight the information that you wish to transfer to the Excel document. Copy that information by selecting "Copy" from the "Edit" menu. You can then paste the information into the Excel document of your choice by opening that document and selecting "Paste" from the "Edit" menu. Information can also be copied from an Excel document into CRUISER by following a similar procedure. Select and copy the desired information from the Excel document then switch to CRUISER and open the "Workbook Designer" screen shown in Figure 8 (by following the procedure detailed above). Paste the information from Excel into the "Workbook Designer" and close the "Workbook Designer," then select "OK" in the window shown in Figure 6.

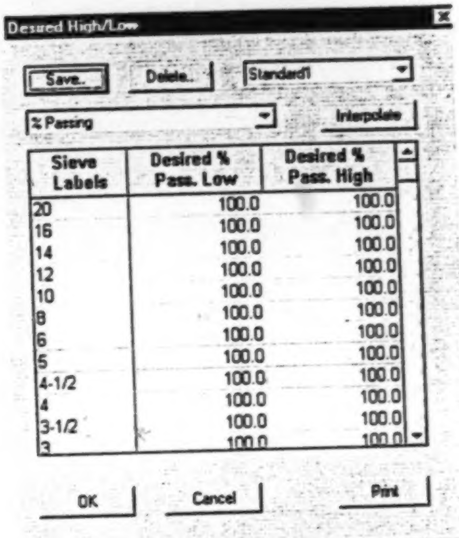

Formula One Workbook Designer
File Edit View Data Sheet Format Object Help
C40
100
For Help, press F1
NUM

Sieve Label	Sieve Size	Sample 1
20	20	0.0
16	16	0.53
14	14	1.05
12	12	1.58
10	10	2.11
8	8	2.63
6	6	3.16
5	5	3.68
4-1/2	4.5	4.21
4	4	4.74
3-1/2	3.5	5.26
3	3	5.79
2-3/4	2.75	6.32
2-1/2	2.5	6.84
2-1/4	2.25	7.37

Figure 13 - Formula One Workbook Designer

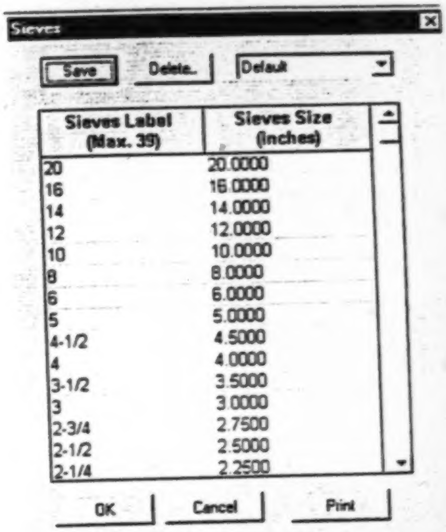
Appendix D

Sample Screens from the Options Menu.


Desired High/Low
Save..
Delete..
Standard1
% Passing
Interpolate
OK
Cancel
Print

Sieve Labels	Desired % Pass. Low	Desired % Pass. High
20	100.0	100.0
16	100.0	100.0
14	100.0	100.0
12	100.0	100.0
10	100.0	100.0
8	100.0	100.0
6	100.0	100.0
5	100.0	100.0
4-1/2	100.0	100.0
4	100.0	100.0
3-1/2	100.0	100.0
3	100.0	100.0

Figure 15 – This screen appears when "Sieves…" is selected from the "Options" Menu.


Sieves
Save
Delete..
Default
OK
Cancel
Print

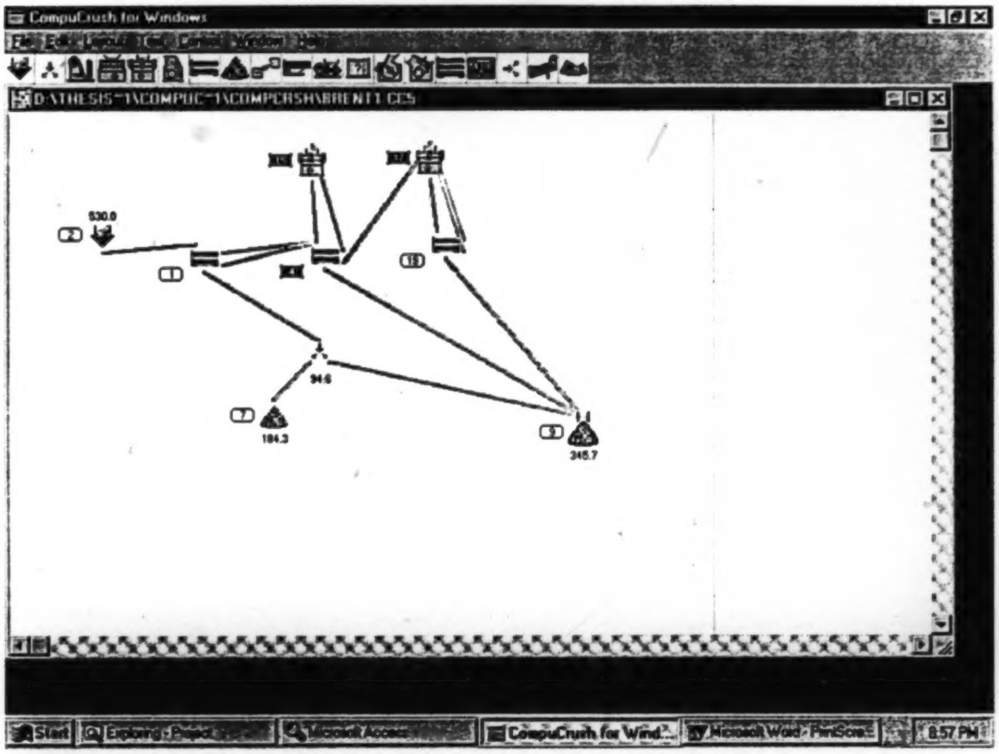
Sieves Label (Max. 39)	Sieves Size (inches)
20	20.0000
16	16.0000
14	14.0000
12	12.0000
10	10.0000
8	8.0000
6	6.0000
5	5.0000
4-1/2	4.5000
4	4.0000
3-1/2	3.5000
3	3.0000
2-3/4	2.7500
2-1/2	2.5000
2-1/4	2.2500

Figure 14 - This screen appears when "Desired..." is selected from the "Options menu.

APPENDIX B

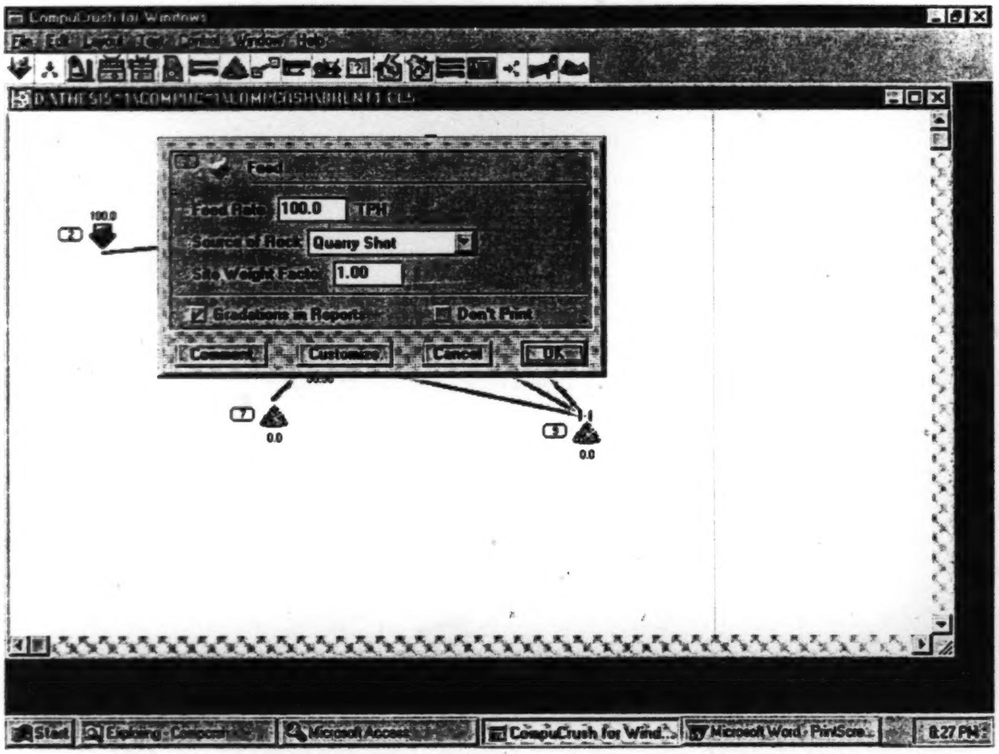
CompuCrush

Plant Layout - CompuCrush



CompuCrush for Windows
File Edit Layout Calc Control Window Help
D:\THESIS~1\COMPUC~1\COMPCRSH\BRENT1.CCS
530.0
2
1
19
94.6
7
184.3
9
345.7
Start
Exploring - Project
Microsoft Access
CompuCrush for Wind...
Microsoft Word - PrntScre...
8:57 PM

Raw Feed Configuration - CompuCrush


CompuCrush for Windows
File Edit Layout Cost Control Window Help
Feed
Comment
Customize
Cancel
OK
100.0
2
7
0.0
9
0.0
Start
Exploring - Compucrush
Microsoft Access
CompuCrush for Wind...
Microsoft Word - PrintScre...
8:27 PM

Feed Rate 100.0 TPH

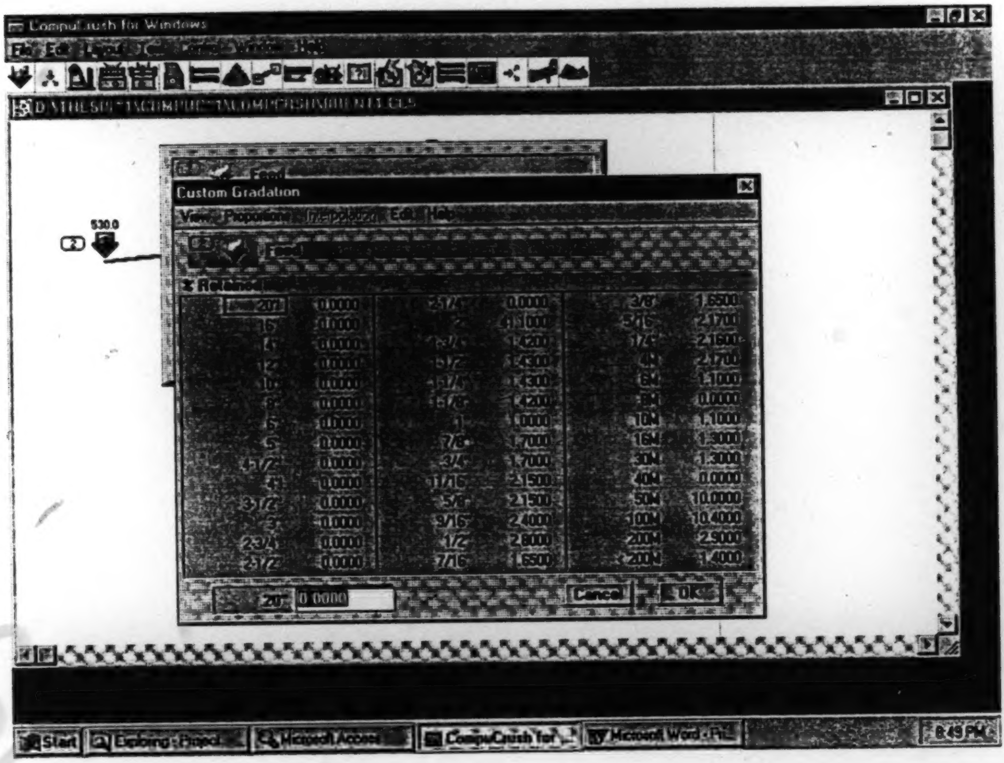
Source of Rock Quarry Shot

Site Weight Factor 1.00

☑ Gradations in Reports

☐ Don't Print

Raw Feed Gradation Input (Chart) - CompuCrush

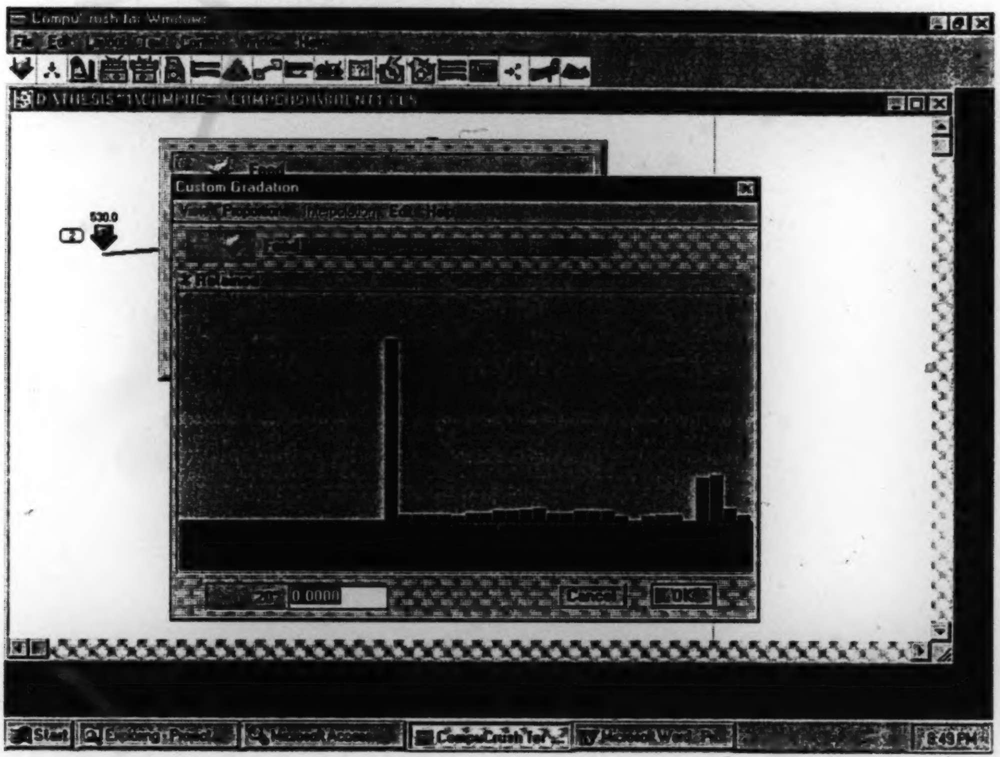

CompuCrush for Windows
Custom Gradation
View
Proportions
Interpolation
Edit
Help
Feed
530.0
Cancel
OK
Start
CompuCrush for...
Microsoft Access
8:49 PM

% Retained

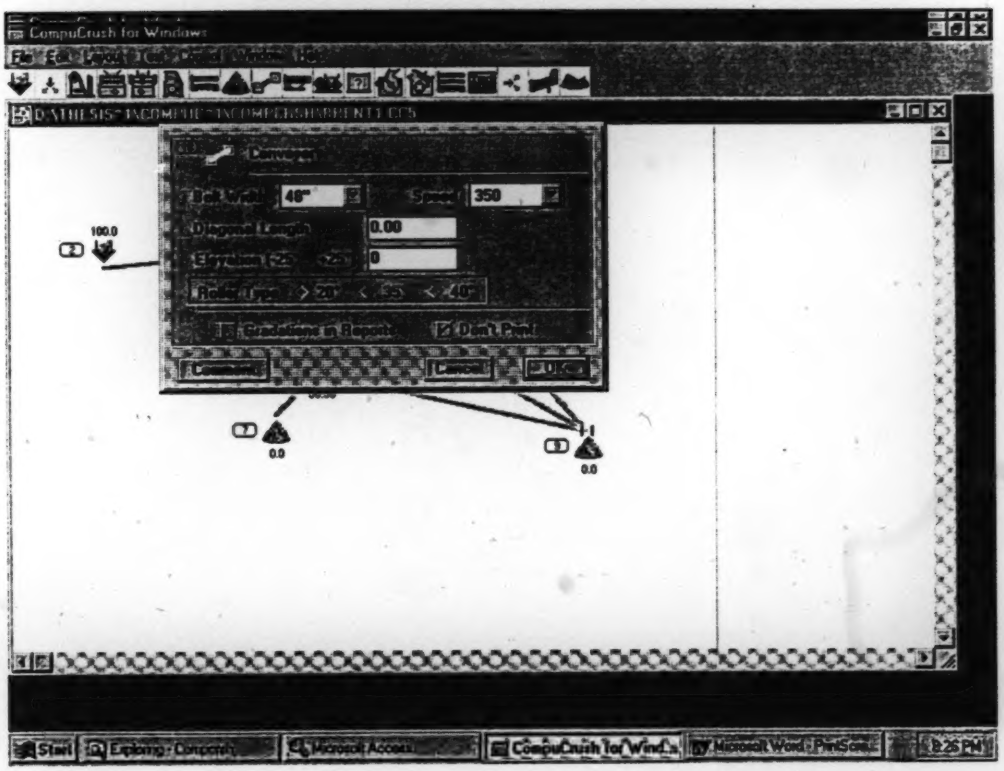
20	0.0000	2-1/4	0.0000	3/8	1.6500
16	0.0000	2	41.1000	5/16	2.1700
14	0.0000	1-3/4	1.4200	1/4	2.1600
12	0.0000	1-1/2	1.4300	4M	2.1700
10	0.0000	1-1/4	1.4300	6M	1.1000
8	0.0000	1-1/8	1.4200	8M	0.0000
6	0.0000	1	1.0000	10M	1.1000
5	0.0000	7/8	1.7000	16M	1.3000
4-1/2	0.0000	3/4	1.7000	30M	1.3000
4	0.0000	11/16	2.1500	40M	0.0000
3-1/2	0.0000	5/8	2.1500	50M	10.0000
3	0.0000	9/16	2.4000	100M	10.4000
2-3/4	0.0000	1/2	2.8000	200M	2.9000
2-1/2	0.0000	7/16	1.6500	<200M	1.4000

20 0.0000

Raw Feed Gradation Input (Graph) - CompuCrush


Custom Gradation
530.0
Feed
0.0000
Cancel
OK
Start
CompuCrush for...

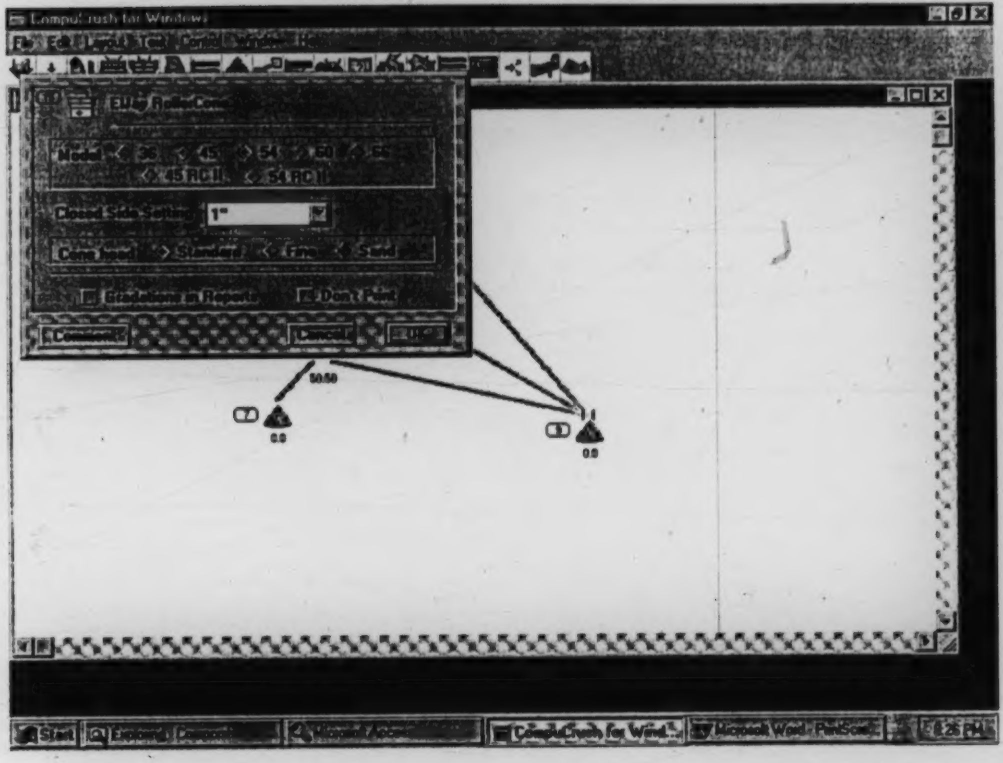
Conveyor Configuration - CompuCrush



Belt Width: 48"
Speed: 350
Diagonal Length: 0.00
Elevation -25 +25: 0

CompuCrush for Windows
File
Edit
Layout
Conveyor
Roller Type
> 20°
< 35
< 40°
Gradations in Reports
☑ Don't Print
Comments
Cancel
OK
100.0
2
7
0.0
8
0.0
Start
Exploring - Compucrush
Microsoft Access
CompuCrush for Wind...
Microsoft Word - PrntScre...
8:26 PM

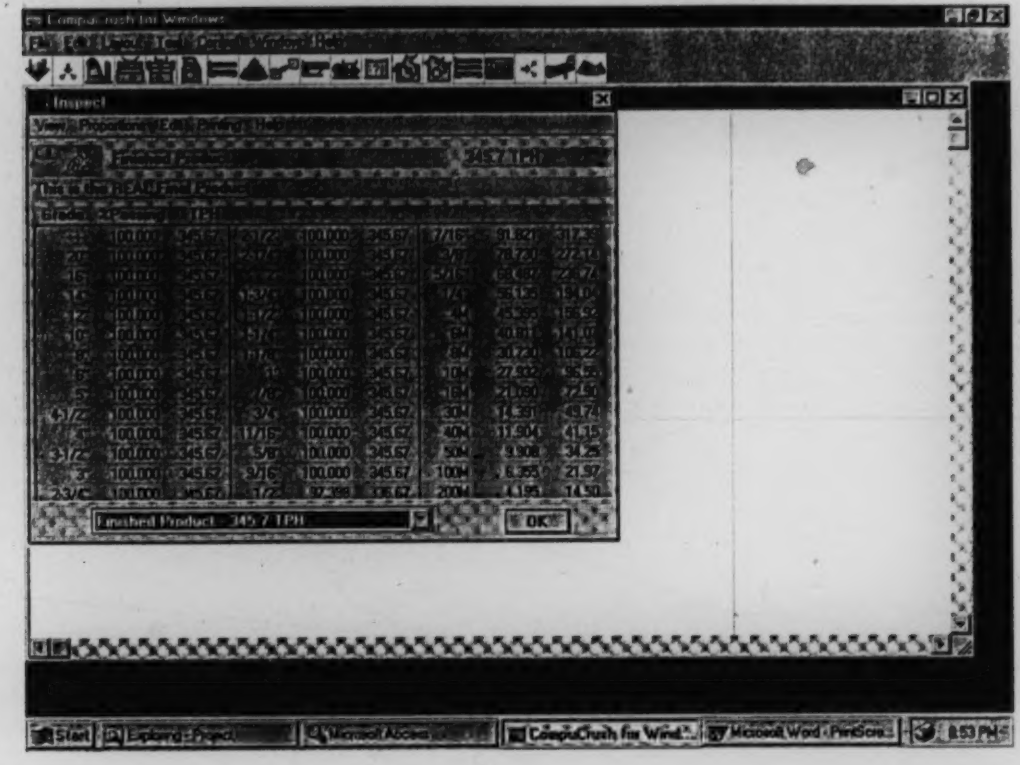
Cone Crusher Configuration - CompuCrush



CompuCrush for Windows
Model
36
45
54
60
66
45 RC II
54 RC II
Cone head
Standard
Fine
Sand
Gradations in Reports
Don't Print
Cancel
50.50
0.0
0.0

Closed Side Setting: 1"

Final Product Output (Chart Format) - CompuCrush


CompuCrush for Windows
File Edit Layout Tools Options Window Help
Inspect
View Proportions Edit Printing Help
OK
Start
Exploring - Project
Microsoft Access
CompuCrush for Wind...
Microsoft Word - PrintScre...
8:53 PM

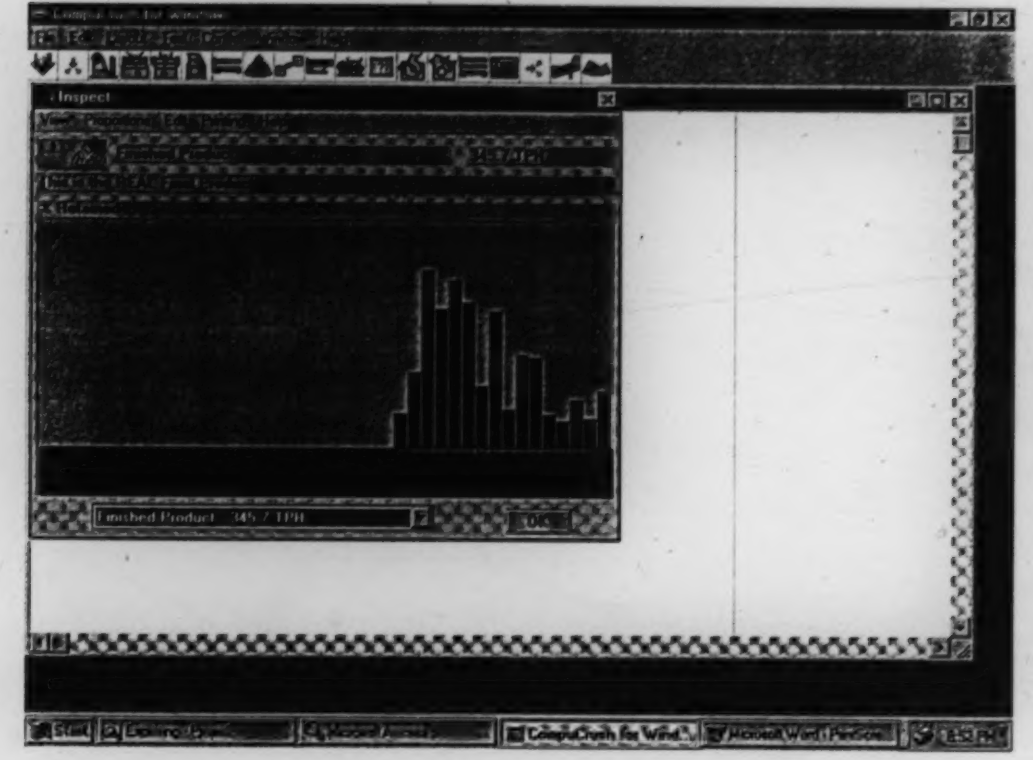
Finished Product 345.7 TPH

This is the REAL Final Product

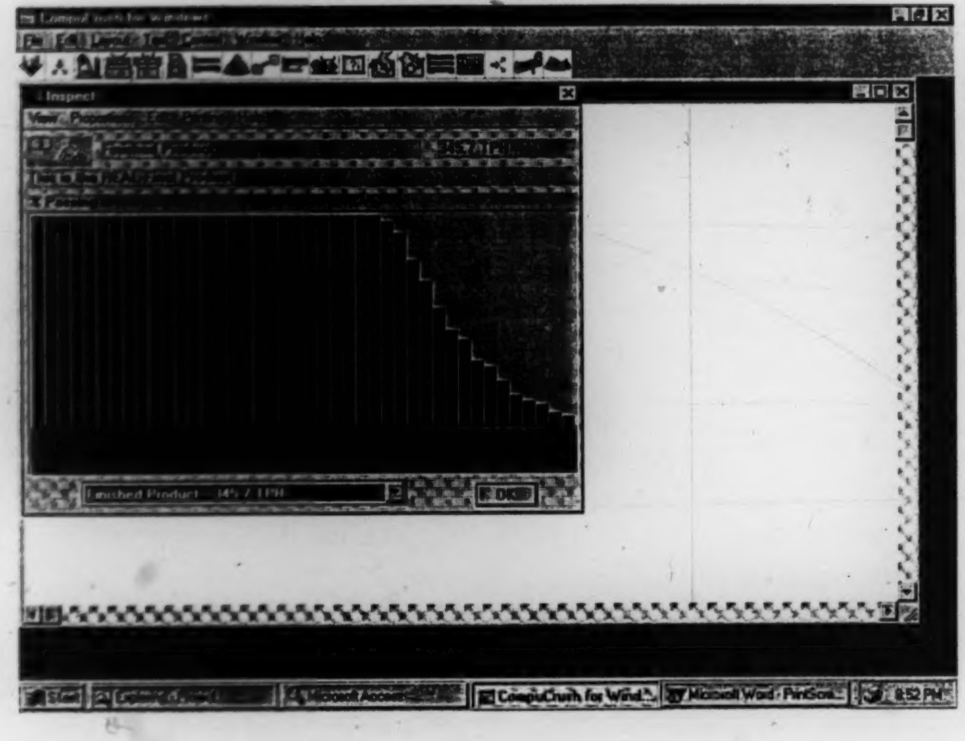
Grade	%Passing	TPH						
[illegible]	100.000	345.67	2-1/2"	100.000	345.67	7/16"	91.821	317.39
20"	100.000	345.67	2-1/4"	100.000	345.67	3/8"	78.730	272.14
16"	100.000	345.67	[illegible]	100.000	345.67	5/16"	68.487	236.74
14"	100.000	345.67	1-3/4"	100.000	345.67	1/4"	56.135	194.04
12"	100.000	345.67	1-1/2"	100.000	345.67	4M	45.395	156.92
10"	100.000	345.67	1-1/4"	100.000	345.67	6M	40.811	141.07
8"	100.000	345.67	1-1/8"	100.000	345.67	8M	30.730	106.22
6"	100.000	345.67	1"	100.000	345.67	10M	27.932	96.55
5"	100.000	345.67	7/8"	100.000	345.67	16M	21.090	72.90
4-1/2"	100.000	345.67	3/4"	100.000	345.67	30M	14.391	49.74
4"	100.000	345.67	11/16"	100.000	345.67	40M	11.904	41.15
3-1/2"	100.000	345.67	5/8"	100.000	345.67	50M	9.908	34.25
3"	100.000	345.67	9/16"	100.000	345.67	100M	6.355	21.97
2-3/4"	100.000	345.67	1/2"	97.398	336.67	200M	4.195	14.50

Finished Product - 345.7 TPH

Final Product Output (Graph Format % Retained) - CompuCrush

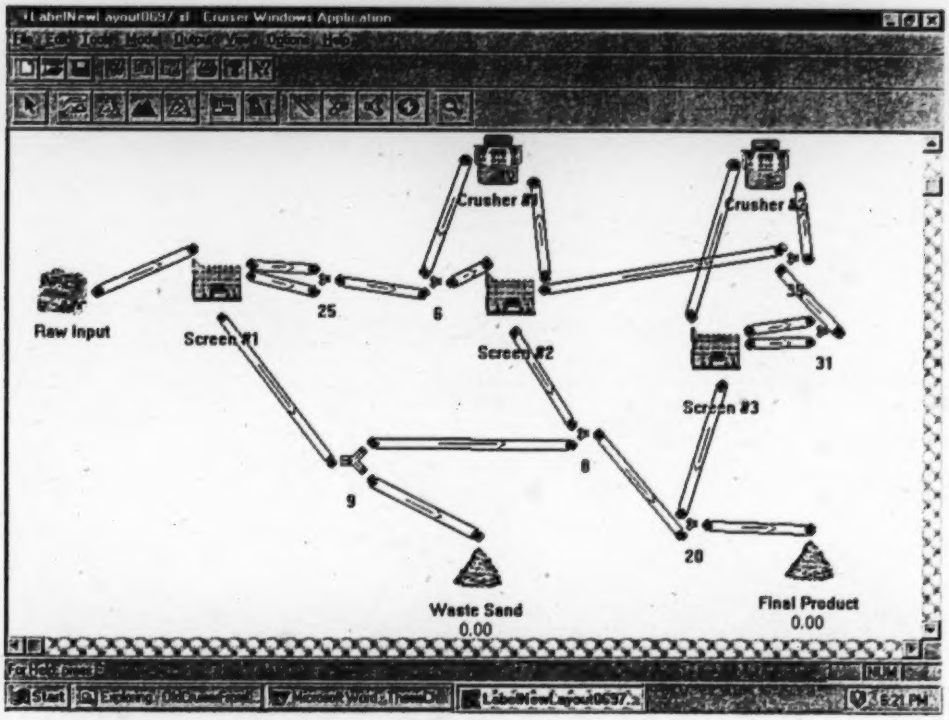

Inspect
Finished Product - 345.7 TPH
OK
Start
CompuCrush for Wind...

Final Output (Graph Format - % Passing) - CompuCrush

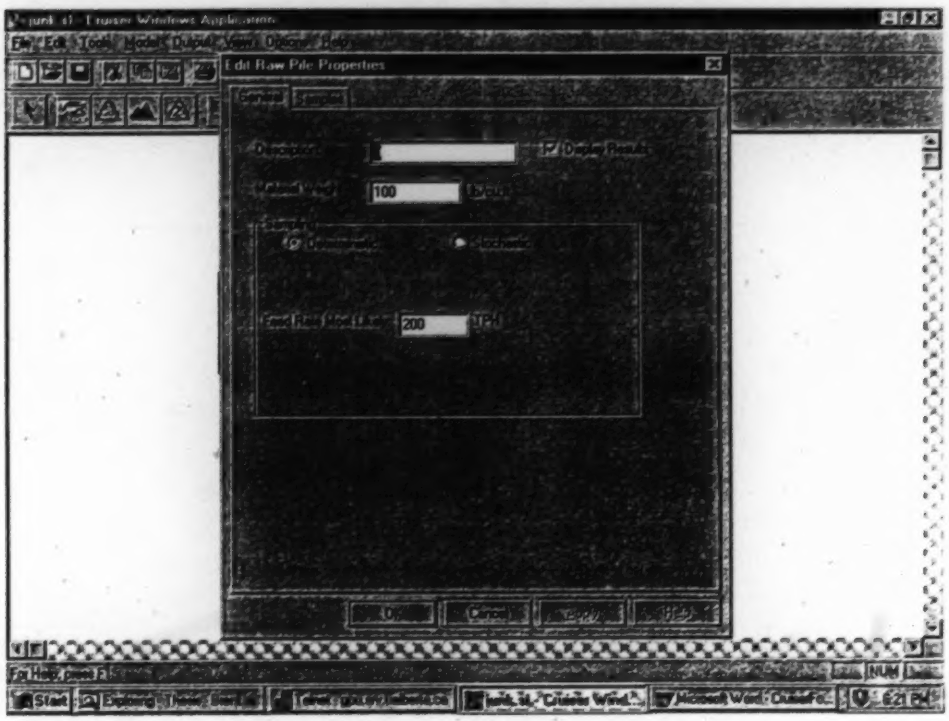

Inspect
% Passing
OK
Start
Microsoft Access
CompuCrush for Wind...
Microsoft Word - PrintScre...
8:52 PM

CRUISER

Plant Layout - CRUISER


LabelNewLayout0697.sf - Cruiser Windows Application
File
Edit
Tools
Model
Output
View
Options
Help
Crusher #1
Crusher #2
Raw Input
Screen #1
25
6
Screen #2
31
Screen #3
9
8
20
Waste Sand
0.00
Final Product
0.00
For Help, press F1
NUM
Start
LabelNewLayout0697.s
6:21 PM

Raw Feed Configuration – CRUISER


Cruiser Windows Application
File Edit Tools Model Output View Options Help
Edit Raw Pile Properties
General
OK
Cancel
Apply
Help
For Help, press F1
NUM
Start
Microsoft Word
6:21 PM

Description:

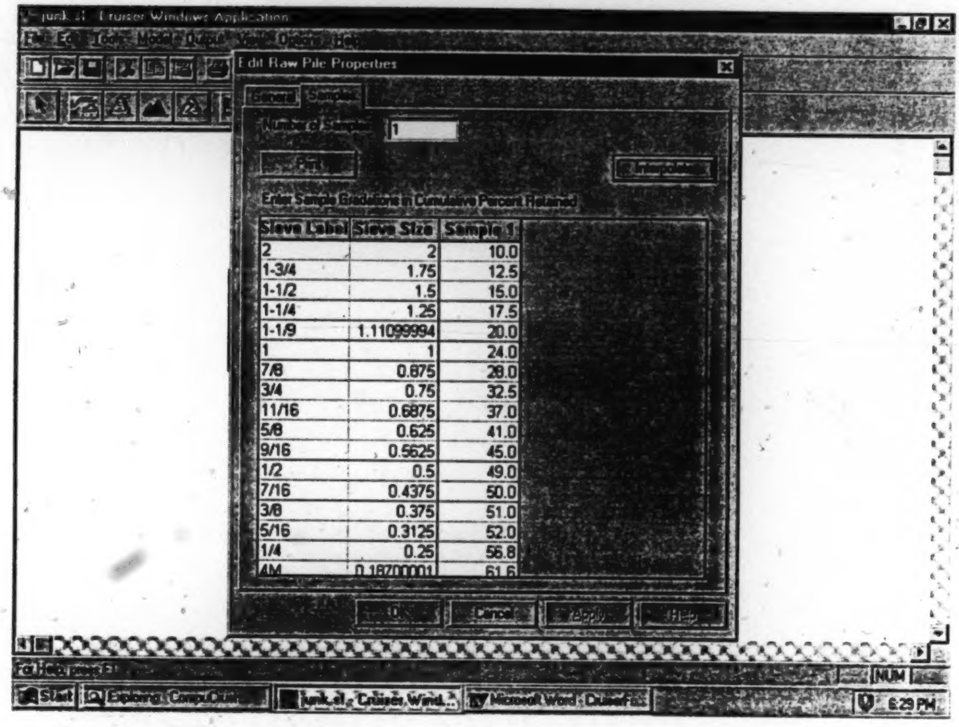
☑ Display Results

100

○ Stochastic

200 TPH

Raw Feed Gradation Input (Chart) - CRUISER

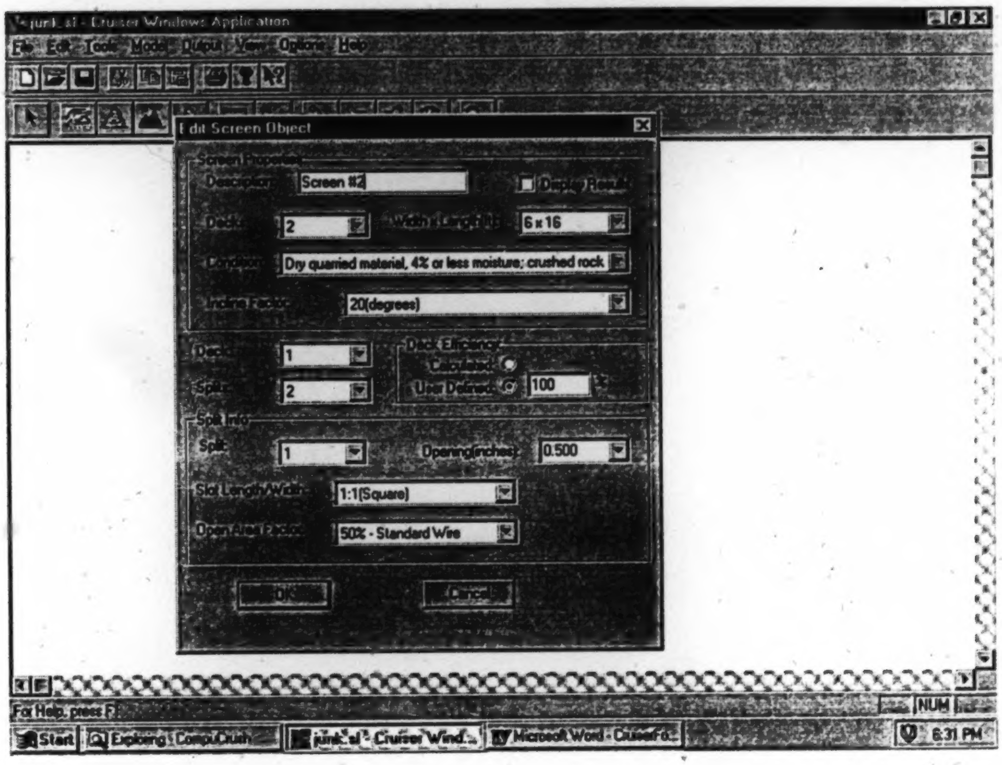

Cruiser Windows Application
Edit Raw Pile Properties
General
Samples
Plot
Interpolate
OK
Cancel
Apply
Help
Start
NUM
6:29 PM

Number of Samples: 1

Enter Sample Gradations in Cumulative Percent Retained

Sieve Label	Sieve Size	Sample 1
2	2	10.0
1-3/4	1.75	12.5
1-1/2	1.5	15.0
1-1/4	1.25	17.5
1-1/9	1.11099994	20.0
1	1	24.0
7/8	0.875	28.0
3/4	0.75	32.5
11/16	0.6875	37.0
5/8	0.625	41.0
9/16	0.5625	45.0
1/2	0.5	49.0
7/16	0.4375	50.0
3/8	0.375	51.0
5/16	0.3125	52.0
1/4	0.25	56.8
4M	0.18700001	61.6

Screen Configuration - CRUISER



junk_sl - Cruiser Windows Application
File Edit Tools Model Output View Options Help
Edit Screen Object
OK
Cancel
For Help, press F1
NUM
Start
Exploring - CompuCrush
junk_sl - Cruiser Wind...
Microsoft Word - CruiserFo...
6:31 PM

Screen Properties

Description: Screen #2

☐ Display Results

Decks: 2

Width x Length (ft): 6 x 16

Condition: Dry quarried material, 4% or less moisture; crushed rock

Incline Factor: 20(degrees)

Deck: 1

Splits: 2

Deck Efficiency

Calculated ○

User Defined ● 100 %

Split Info

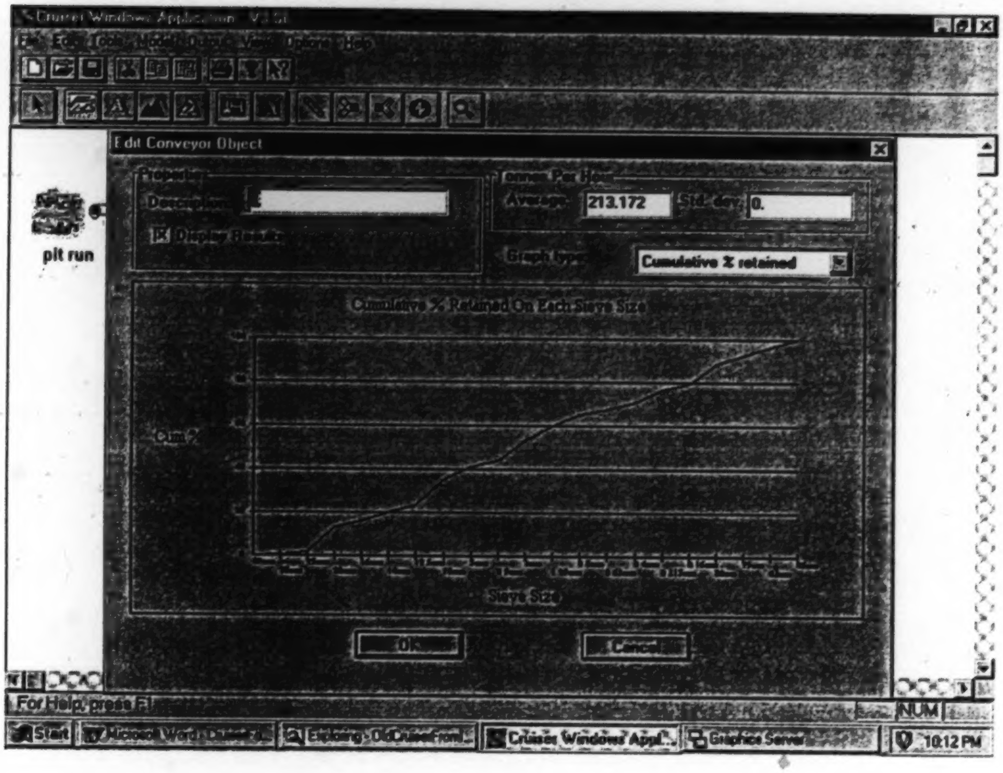
Split: 1

Opening(inches): 0.500

Slot Length/Width: 1:1(Square)

Open Area Factor: 50% - Standard Wire

Conveyor Output - CRUISER



Tonnes Per Hour

Average: 213.172

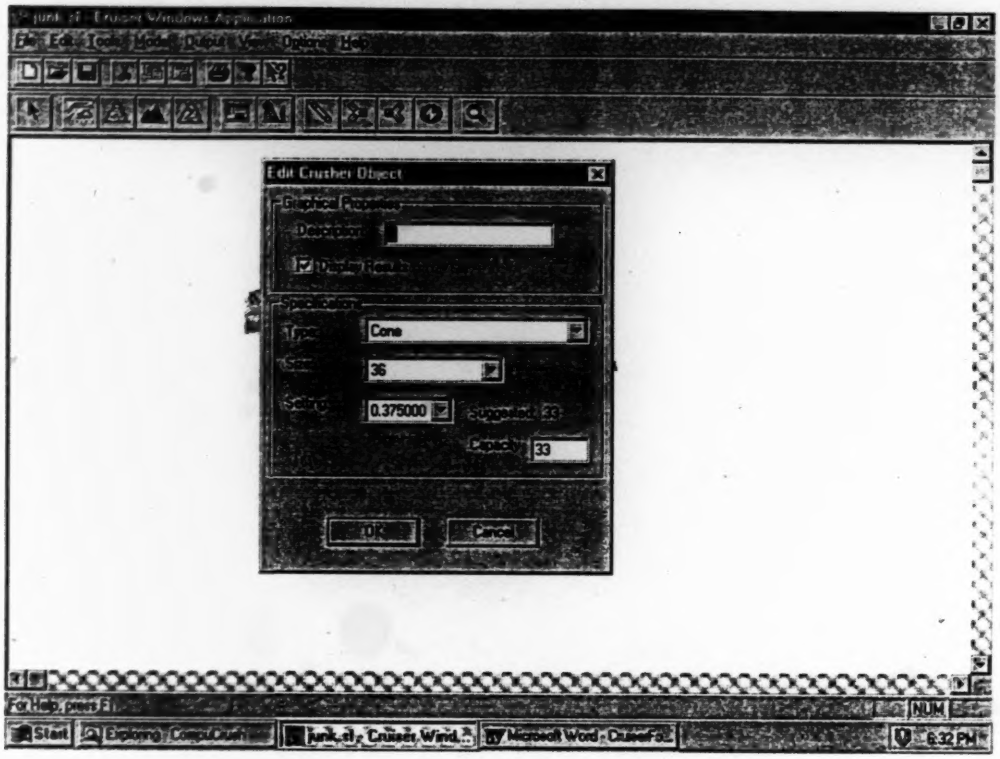
Std. dev: 0.

☒ Display Results

Graph type: Cumulative % retained

Cruiser Windows Application - V3.51
File Edit Tools Models Outputs View Options Help
Edit Conveyor Object
Properties
Description
pit run
Cumulative % Retained On Each Sieve Size
Cum %
Sieve Size
OK
Cancel
For Help, press F1
NUM
Start
Microsoft Word - Dunedin...
Exploring - OldCruiserFront...
Cruiser Windows Appl...
Graphics Server
10:12 PM

Cone Crusher Configuration - CRUISER


junk.sf - Cruiser Windows Application
File Edit Tools Model Output View Options Help
Edit Crusher Object
Graphical Properties
Specifications
OK
Cancel
For Help, press F1
NUM
Start
Exploring - CompuCrush
junk.sf - Cruiser Wind...
Microsoft Word - CruiserFo...
6:32 PM

Description:

☑ Display Results

Type: Cone

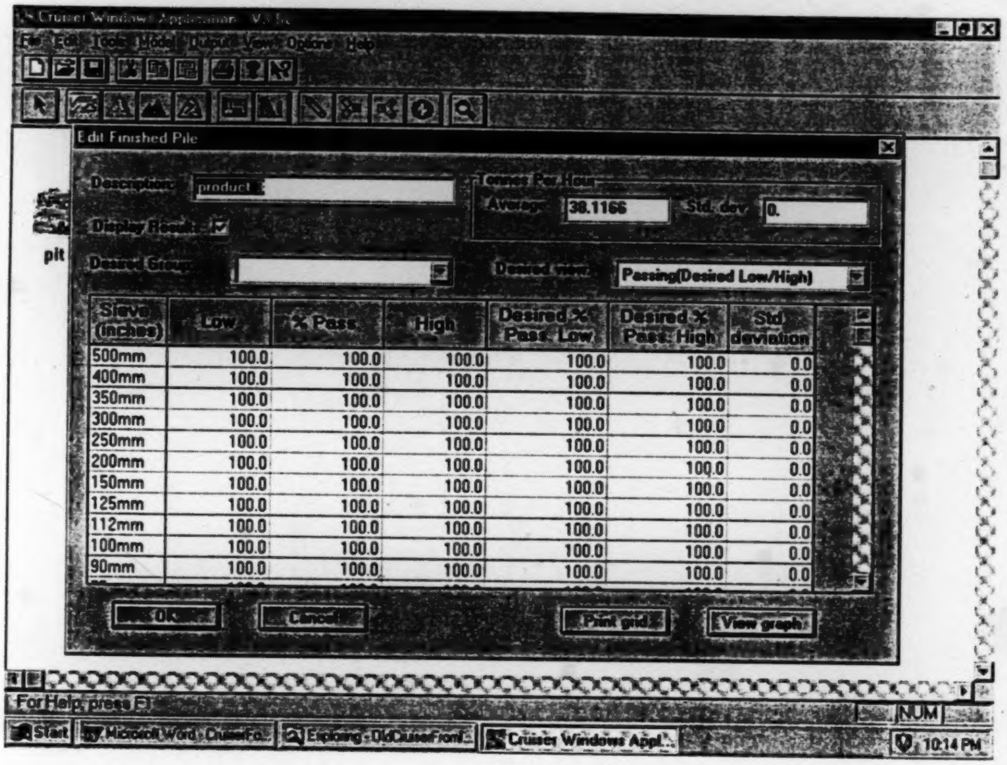
Size: 36

Setting: 0.375000

Suggested: 33

Capacity: 33

Final Product Output (Chart Format) - CRUISER


Cruiser Windows Application - V1.5c
File Edit Tools Model Output View Options Help
Edit Finished Pile
pit
OK
Cancel
Print grid
View graph
For Help, press F1
NUM
Start
Microsoft Word - CruiserFo...
Exploring - OldCruiserFront...
Cruiser Windows Appl...
10:14 PM

Description: product

Display Results: ☑

Desired Group:

Tonnes Per Hour

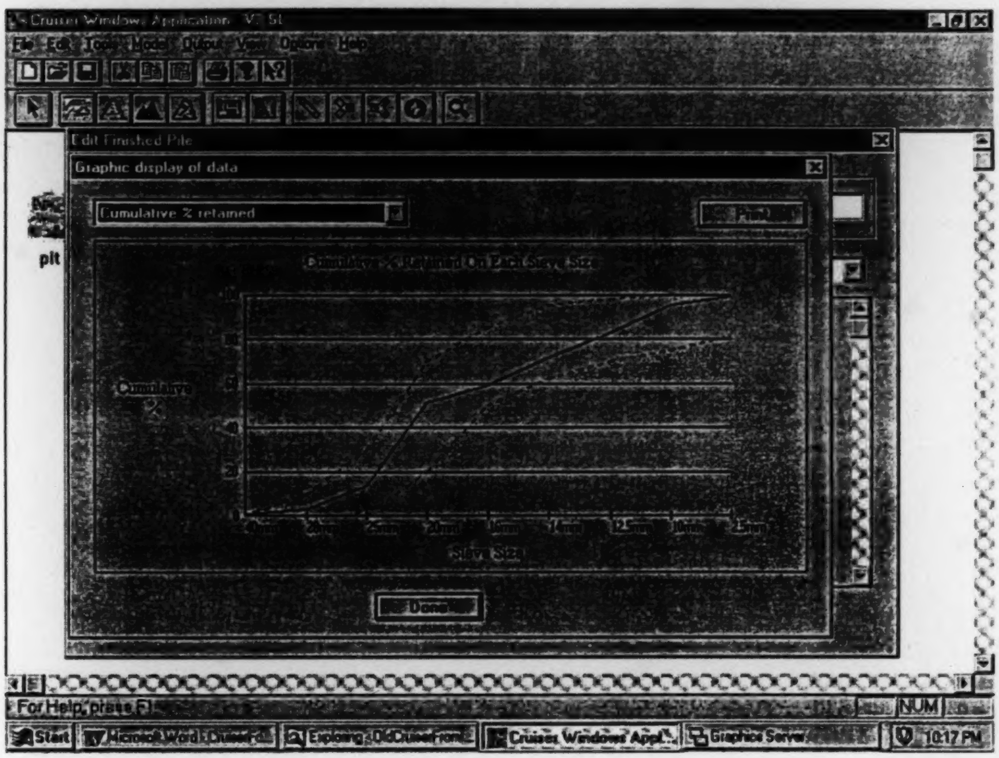
Average: 38.1166

Std. dev: 0.

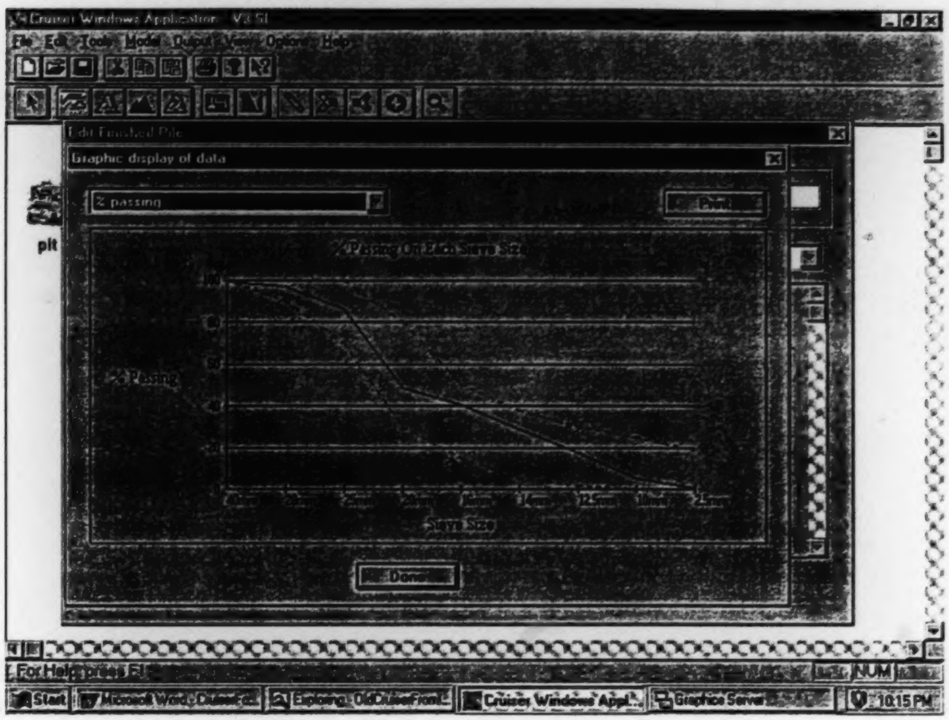
Desired view: Passing(Desired Low/High)

Sieve (inches)	Low	% Pass	High	Desired % Pass Low	Desired % Pass High	Std deviation
500mm	100.0	100.0	100.0	100.0	100.0	0.0
400mm	100.0	100.0	100.0	100.0	100.0	0.0
350mm	100.0	100.0	100.0	100.0	100.0	0.0
300mm	100.0	100.0	100.0	100.0	100.0	0.0
250mm	100.0	100.0	100.0	100.0	100.0	0.0
200mm	100.0	100.0	100.0	100.0	100.0	0.0
150mm	100.0	100.0	100.0	100.0	100.0	0.0
125mm	100.0	100.0	100.0	100.0	100.0	0.0
112mm	100.0	100.0	100.0	100.0	100.0	0.0
100mm	100.0	100.0	100.0	100.0	100.0	0.0
90mm	100.0	100.0	100.0	100.0	100.0	0.0

Final Product Output (Graph Format – Cumulative Retained) - CRUISER

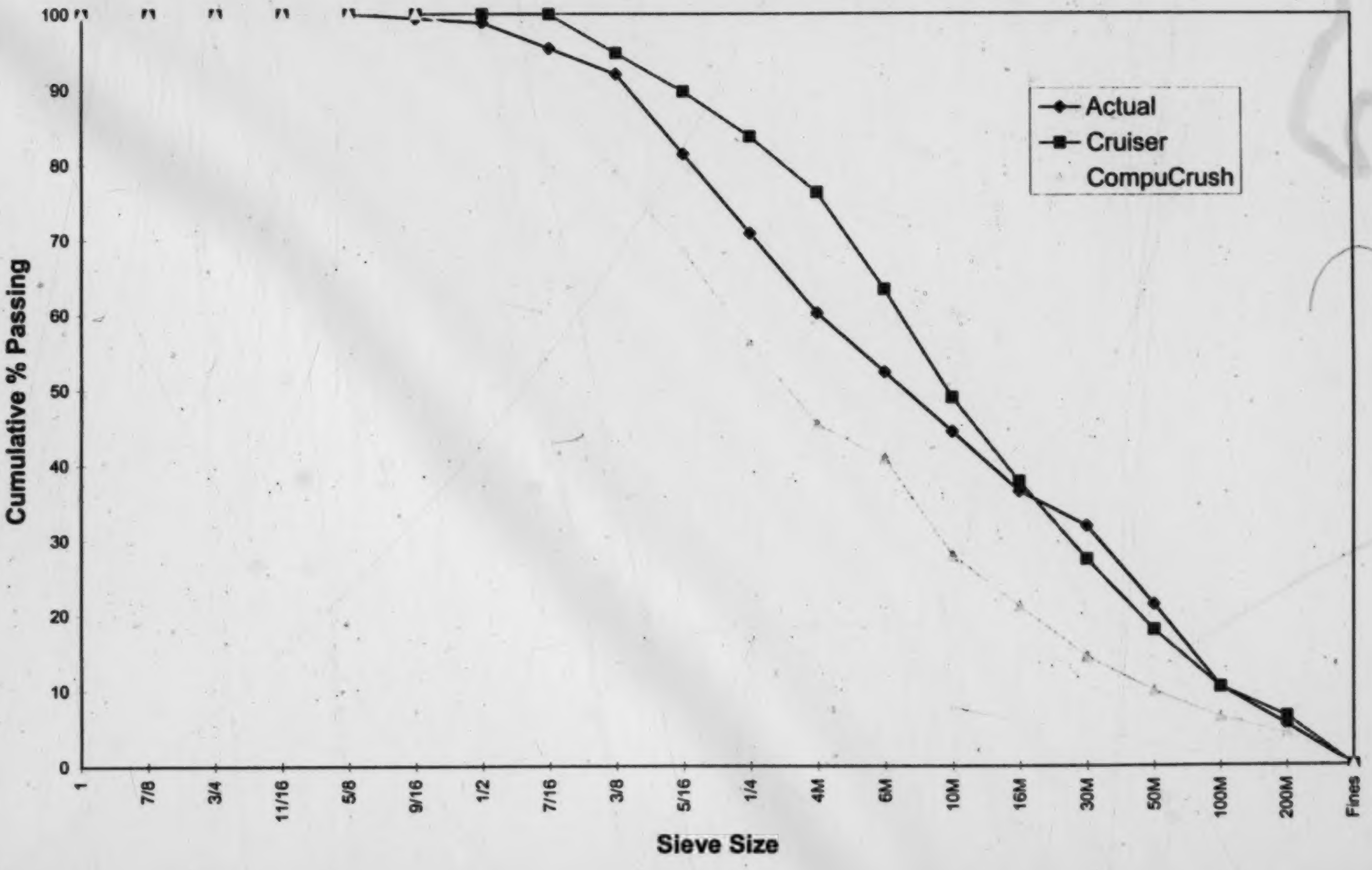

Cruiser Windows Application V2.51
File Edit Tools Model Output View Options Help
Edit Finished Pile
Graphic display of data
Cumulative % retained
Print
plt
Cumulative % Retained On Each Sieve Size
Cumulative %
0
20
40
60
80
100
40mm
28mm
25mm
20mm
16mm
14mm
12.5mm
10mm
2.5mm
Sieve Size
Done
For Help, press F1
NUM
Start
Microsoft Word - CruiserFi...
Exploring - OldCruiserFiona
Cruiser Windows Appl...
Graphics Server
10:17 PM

Final Product Output (Graph Format - % Passing) - CRUISER

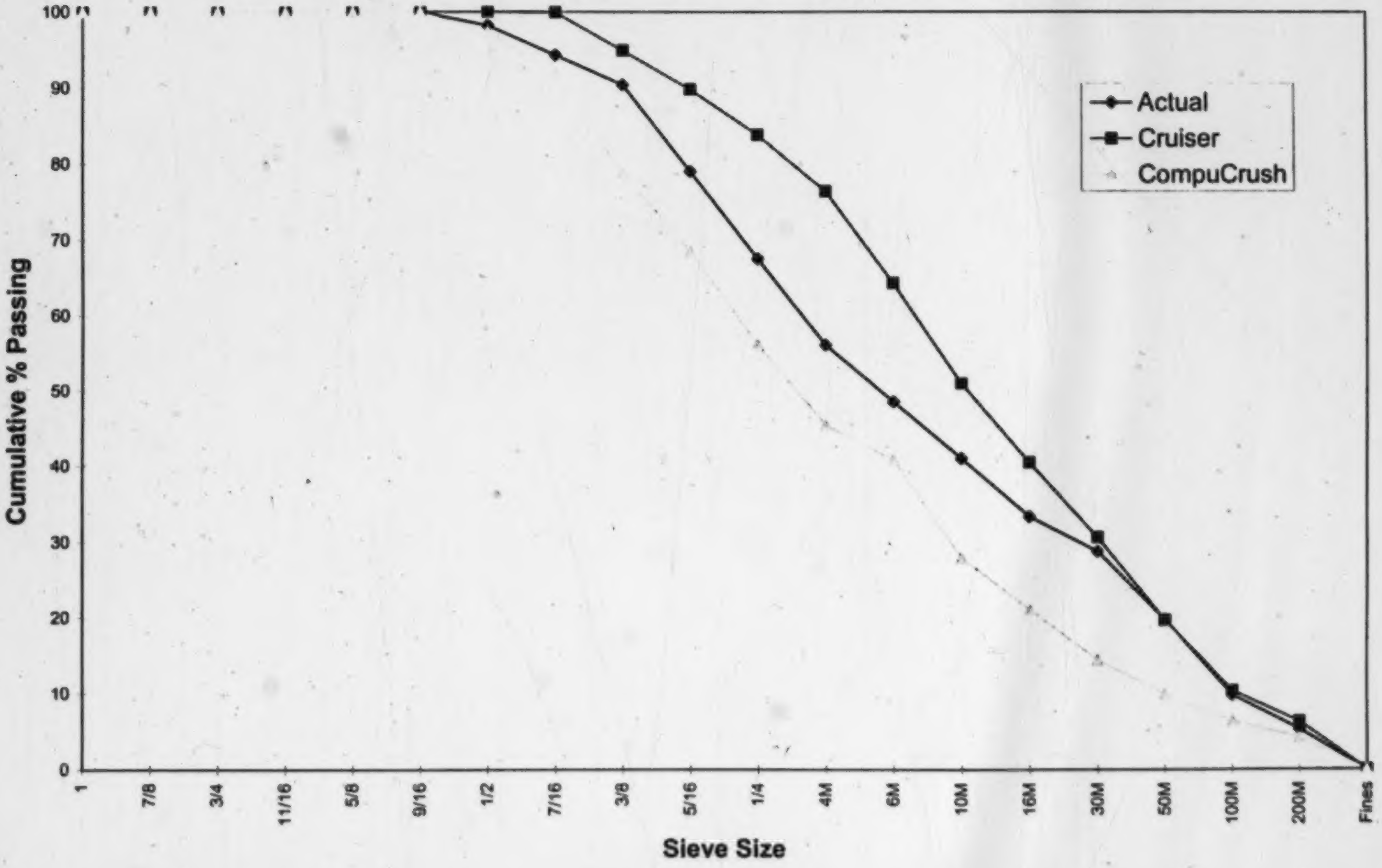

Cruiser Windows Application V3.51
File Edit Tools Model Output View Options Help
Edit Finished Pile
Graphic display of data
% passing
Print
pit
% Passing On Each Sieve Size
100
80
60
40
20
0
% Passing
12.5mm
Sieve Size
Done
For Help, press F1
NUM
Start
Microsoft Word
Exploring - OldCruiserFront
Cruiser Windows Appl...
Graphics Server
10:15 PM

Gradation Comparison

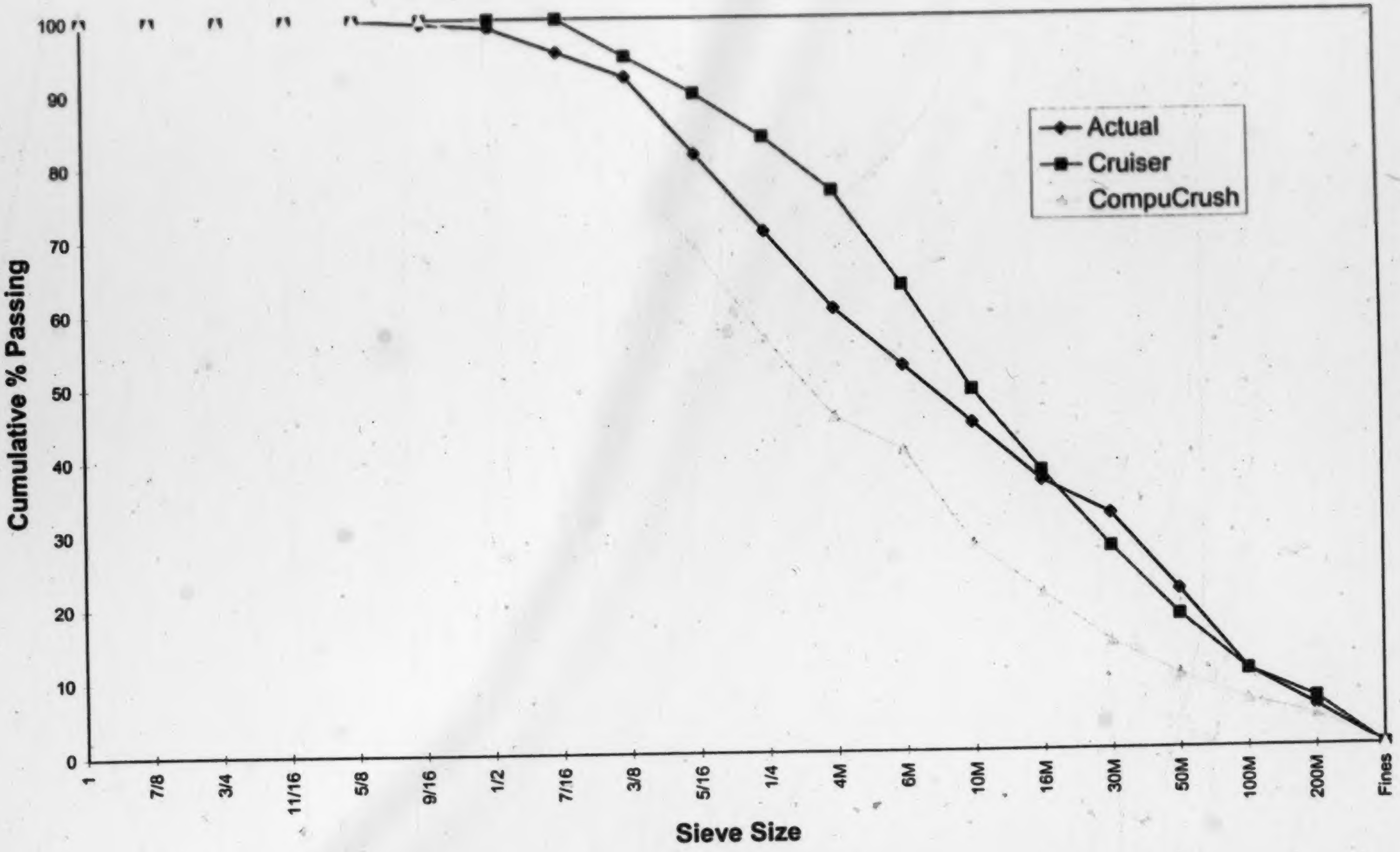
Comparison Between Actual, Cruiser, and CompuCrush Gradations - ACO Sample #1

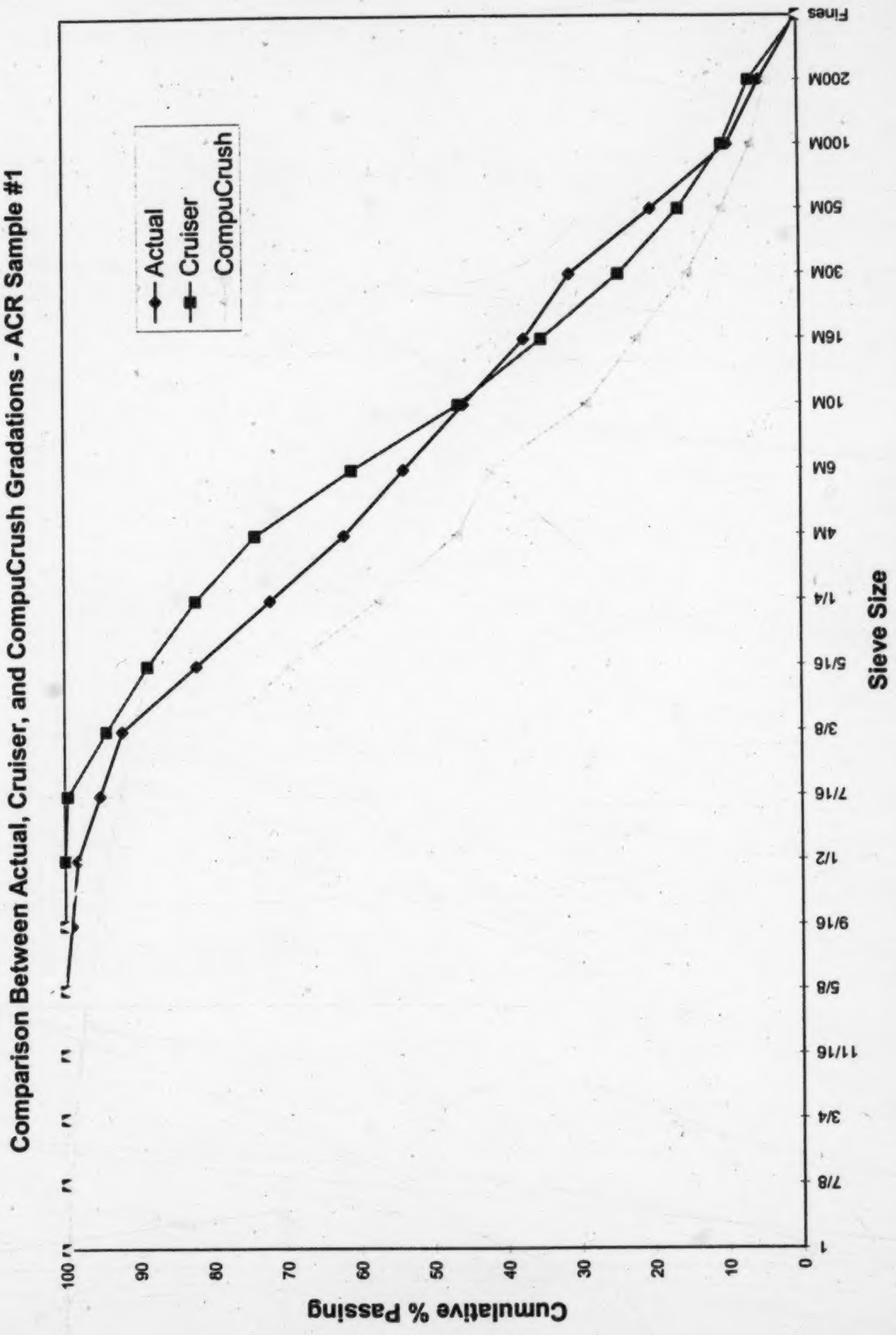


Comparison Between Actual, Cruiser, and CompuCrush Gradations - ACO Sample #2



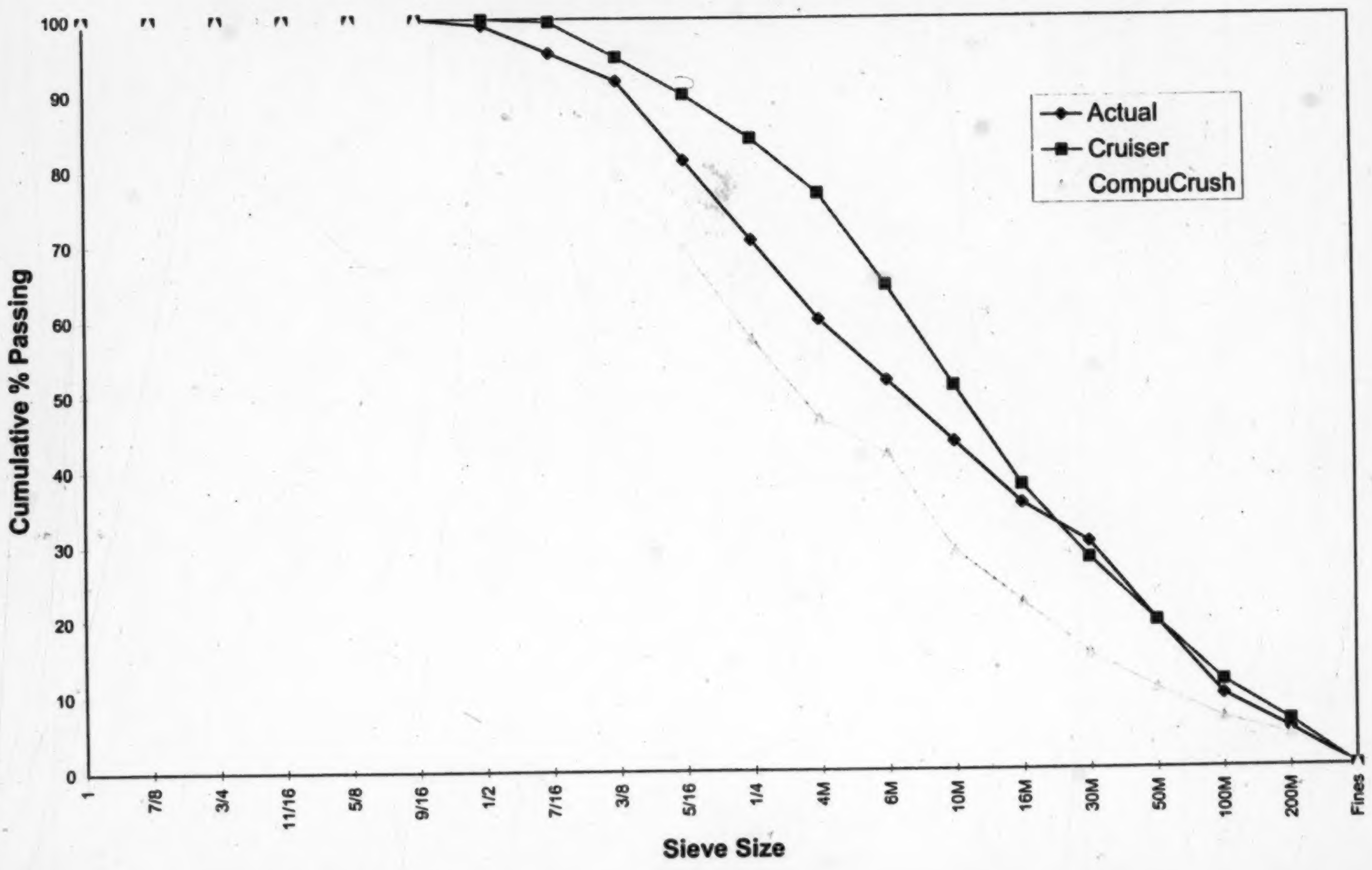
Comparison Between Actual, Cruiser, and CompuCrush Gradations - ACO Sample #3

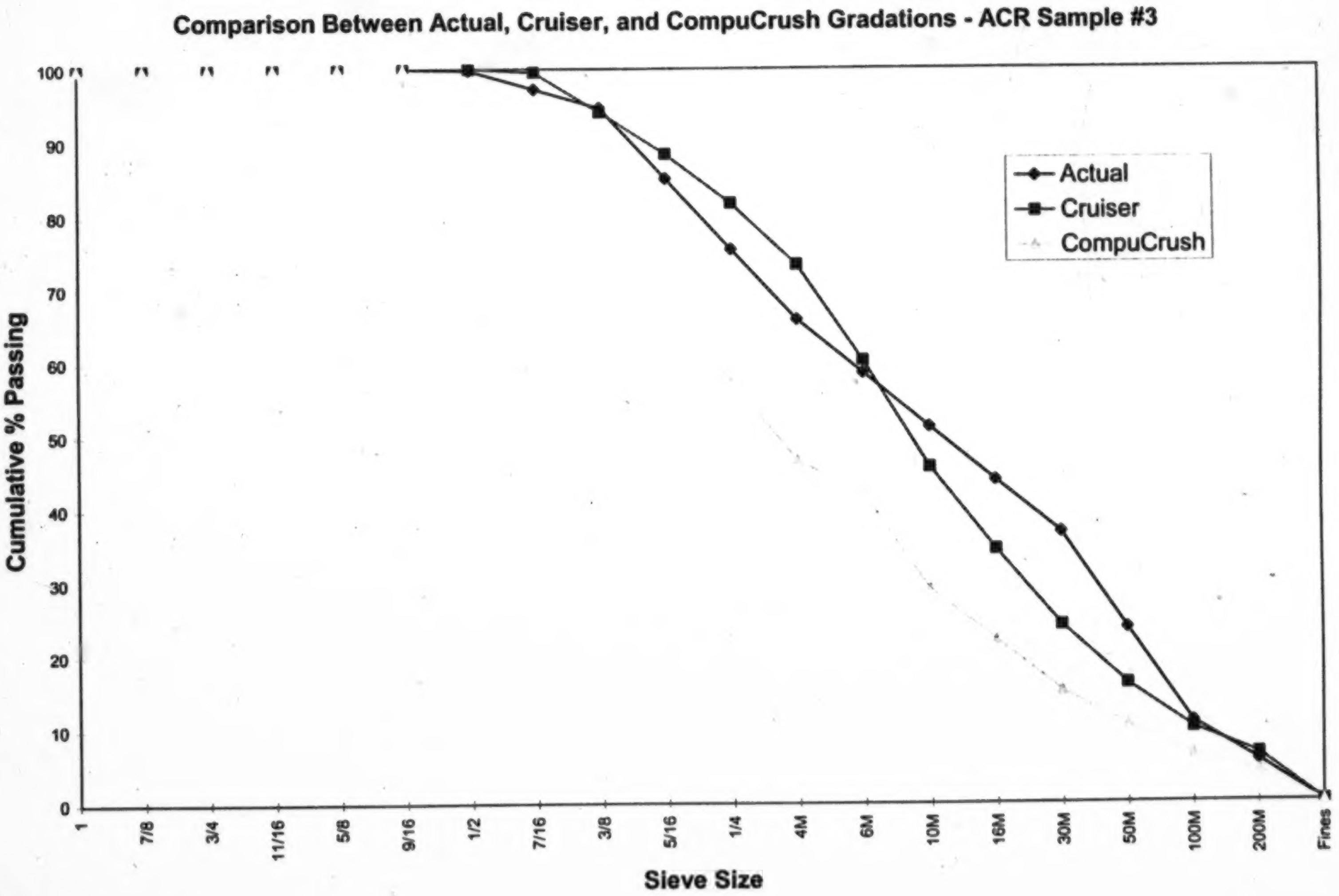




Comparison Between Actual, Cruiser, and CompuCrush Gradations - ACR Sample #1

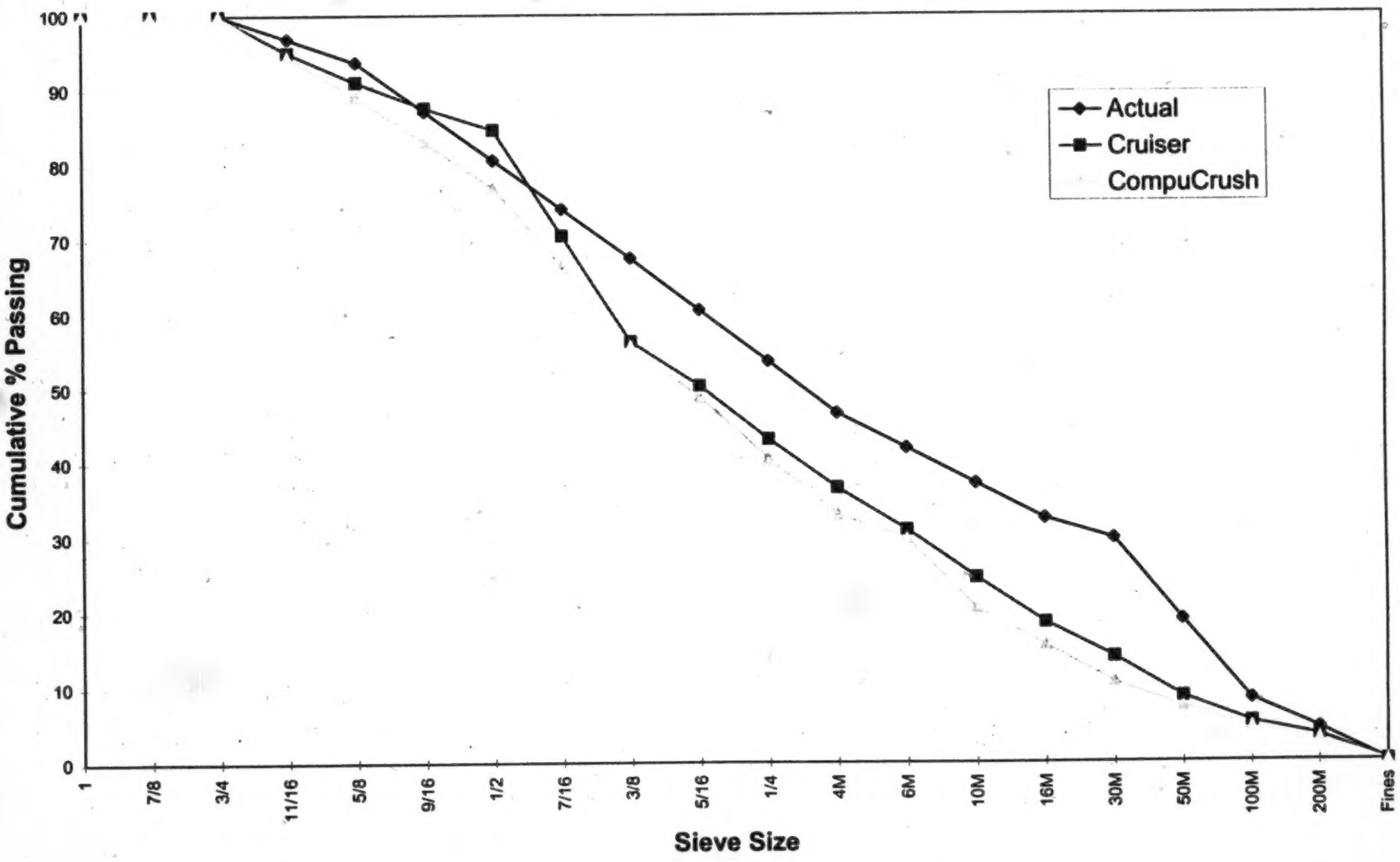
Comparison Between Actual, Cruiser, and CompuCrush Gradations - ACR Sample #2





Comparison Between Actual, Cruiser, and CompuCrush Gradations - ACR Sample #3

Comparison Between Actual, Cruiser, and CompuCrush Gradations - 20mm Road Crush
Sample #1



APPENDIX C

CEM

University of Alberta

Analysis of Crushing Plant Operations

Data Collection Handbook

COMPANY : **LOCATION :**

DATE : **ARTIST :**

SITE LAYOUT

Provide an accurate diagram of the site layout and label each equipment - for conveyers use numerical labels. Repeat the same drawing on the next page for the tester. Make sure you label required test points.

Site Layout (Tester's copy)

Required Information For Samples:

1. Source of material (refer to labels in the diagram)
2. Date and time.
3. Gradation using as many sizes as possible.
4. Moisture content.
5. Density (example 100 lbs/cuft).
6. Weather Conditions (normal, rain,…)

CRUSHER INFORMATION

Label (From diagram)	Type	Manufac turer	Age	Serial #	Setting	Speed	Head & Liner Type	Age of liners

SCREEN INFORMATION

Fill in screen information into tables provided on next two pages. Use the following guidelines:

A : Inclination - Possible values are
1) Horizontal, High Speed Sand, 2) Horizontal, Low Amplitude Stroke, 3) Horizontal, Normal Amplitude Stroke, 4) 5°, 5) 10°, 6) 15°, 7) 20°, 8) 25°, or 9) 30°
B: Material Condition - Possible values are :
1) Moist or dirty stone, 2) Moist ore from underground; coal, 3) Dry quarried material, 4% or less moisture; crushed rock and gravel, or 4) Dry uncrushed material, 6% or less moisture; hot dry material from drier; gravel - clean, not cemented; wet screening with sprays, 1'' material.
C: Slot Length/Width Ratio- Possible values are
1) 2:1, 3) 3:1, 4) 4:1, 5) 5:1, 6) 6:1, 7) more than 6:1, 8) Square, or 9) Round
D: Width x Length -
E: Opening Size -
F: Percent Open Area - Possible values are the following:
1) 30% - Very Heavy Wire, 2) 40% - Heavy Wire, 3) 50% - Standard Wire, 4) 60% - Light Wire, or 4) 70% - Very Light Wire

Label (From diagram)	A	B	Deck	Split	C	D	E	F
			1	1				
				2				
				3				
			2	1				
				2				
				3				
			3	1				
				2				
				3				
			1	1				
				2				
				3				
			2	1				
				2				
				3				
			3	1				
				2				
				3				
			1	1				
				2				
				3				
			2	1				
				2				
				3				
			3	1				
				2				
				3				

CONVEYER INFORMATION

Label (From diagram)	Length- Axle to Axle (feet)	Axle diameter (inches)	Time for Five revolutions (min)	Production Rate (tph)

CEM

Site Report

Date Collected : ________________
Date Tested : ________________
Tester : ________________

Temperature : ________________

To Do:

CHANGES IN OPERATIONS

Verify all equipment information. Comment on any changes in site layout, crusher settings, screen sizes, etc. Verify dates of changes.

Material Information

Loader Analysis. 10 minute study information:

Dump Number	Bucket Capacity	Dump Number	Bucket Capacity
1		6	
2		7	
3		8	
4		9	
5		10	

Struck Bucket Capacity:__________ Time at last dump: ____________

Production rate according to loader time studies: ____________

Production Rate Analysis

Production rate according to weight scale: (1 min intervals):______________________

Average Rate(tph): _________

Conveyer Analysis

Conveyer Label	Length of Strip (feet)	Sample Pale Weight (lb)	Volumetric Pale Weight (lb)	Distance From Top (cm)	Density (lb/ft3)	TPH Calculated
Pitrun						
Sand						
Screened Pitrun						
Coarse Feed						
Coarse Return						
Fine Feed						
Fine Return						
Product						

Volumetric Pail:

weight (empty): 25 lb height = 0.869 feet

diameter = 0.8399 feet volume = 0.48169 cubic feet

Regular Pail weight (empty): 2.5 lb

SIEVE TESTS

VERIFY THAT ALL REQUIRED INFORMATION HAS BEEN SUPPLIED

Number of samples: ___

Sample locations: ____________________________

APPENDIX D

TPH CALCULATIONS		Trial #2 Thursday, October 3, 1996			Product :	ACO				Factor
							weekly input		Weekly outpu	0.847036943
#1 - AGGREGATE SAMPLE								50 cm = 1.64 ft		ADJUSTED
Conveyor Label	Description	Length (ft) (axel to axle)	Axle Diam. (Inches)	Length Per Revolution(ft)	Time Per Revolution(sec.)	# Of Belt Lengths/hr	Sample Pail Weight(lbs)	Mass Per 50cm Width	Tonnes Per Hour	Tonnes Per Hour
1	Pitrun	100.333	18	205.4	22.4	160.7	56	53.5	538.2	455.9
2	Sand	47.583	14	98.8	11.6	310.3	26	23.5	219.7	186.1
3	Screened Pit Run	25	12	53.1	10.2	352.9	48	45.5	260.1	220.3
4	Coarse Feed	39.5	10	81.6	19	189.5	76	72.5	341.7	289.5
5	Coarse Return	43	12	89.1	15.6	230.8	60	57.5	360.5	305.4
6	Fine Feed	47.5	12	98.1	19.2	187.5	36	33.5	187.9	159.2
7	Fine Return	40.75	12	84.6	22	163.6	52	49.5	209.0	177.0
8	Product	47.5	12	98.1	23.4	153.8	83	80.5	370.5	313.8
Production Rate According To Weigh Scales		Number Of Samples =		5			Average Weight From Scale:		313.8	
(at one minute intervals in TPH)										
312	310	320	310	317	0	0	0	0	0	
AVERAGE TPH =	313.8	Compared to [illegible]	370.5	TPH from weighed sample						
		Difference of [illegible]	[illegible]							
#2 - LOADER ANALYSIS										
Bucket Capacity =	8.2	cubic yards								
Dump Number	1	2	3	4	5	6	7	8	9	10
Bucket Fill Factor	1.00	1.00	1.15	1.15	1.10	1.10	0.00	0.00	0.00	0.00
Volume (yds3)	8.2	8.2	9.43	9.43	9.02	9.02	0	0	0	0
Total Volume (yds3)	53.3									
Total Cycle Time	10	minutes								
TPH (based on density)=	502.97	TPH	Compared to	538.2 [illegible]	TPH from weighed sample					
			Difference of	7%						
DENSITY CALCUCATIONS										
		Pail Volume (ft3) =	0.48169	diameter =	0.8399	ft				
		Pail Weight =	25 lbs	height =	0.869	ft				
		weekly input		weekly input						
Conveyor Label	Description	Volumetric Pail Weight(lbs)	Tared Sample Weight (lbs)	Distance From Top If Not Full(cm)	Vol. To Use If Not Full	Density (lb/ft3)				
1	Pitrun	79	54	1	0.4635	116.502				
2	Sand	49	24	13.2	0.2417	99.276				
3	Screened Pit Run	70	45	6.3	0.3672	122.558				
4	Coarse Feed	75	50	0	0.4817	103.801				
5	Coarse Return	77	52	0	0.4817	107.953				
6	Fine Feed	60	35	7.7	0.3417	102.422				
7	Fine Return	74	49	0	0.4817	101.725				
8	Product	68	43	2.9	0.4290	100.239				

TPH CALCULATIONS		Trial #3 Tuesday, October 8, 1996			Product :	ACO		[illegible]		Factor [illegible]
							weekly input		Weekly outpu	0.856772644
#1 - AGGREGATE SAMPLE								50 cm = 1.64 ft		ADJUSTED
Conveyor Label	Description	Length (ft) (axel to axle)	Axle Diam. (inches)	Length Per Revolution(ft)	Time Per Revolution(sec.)	# Of Belt Lengths/hr	Sample Pail Weight(lbs)	Mass Per 50cm Width	Tonnes Per Hour	Tonnes Per Hour
1	Pitrun	100.333	18	205.4	22.4	160.7	55	52.5	528.2	452.5
2	Sand	47.583	14	98.8	11.6	310.3	24	21.5	201.0	172.2
3	Screened Pit Run	25	12	53.1	10.2	352.9	60	57.5	328.7	281.6
4	Coarse Feed	39.5	10	81.6	19	189.5	69	66.5	313.5	268.6
5	Coarse Return	43	12	89.1	15.6	230.8	48	45.5	285.3	244.4
6	Fine Feed	47.5	12	98.1	19.2	187.5	43	[illegible]	241.2	206.6
7	Fine Return	40.75	12	84.6	22	163.6	50	[illegible]	211.1	180.8
8	Product	47.5	12	98.1	23.4	153.8	85	[illegible]	391.2	335.2
Production Rate According To Weigh Scales			Number Of Samples =	20			Average Weight From Scale:		335.15	
(at one minute intervals in TPH)										
1	2	3	4	5	6	7	8	9	10	
341	330	334	337	340	343	327	328	340	355	
11	12	13	14	15	16	17	18	19	20	
305	340	350	310	330	333	342	340	325	353	
AVERAGE TPH =	335.15	As Compared To [illegible] 391.2 [illegible]		TPH from weighed sample						
		Difference of [illegible] 17 [illegible]								
#2 - LOADER ANALYSIS										
Bucket Capacity =	9	cubic yards								
Dump Number	1	2	3	4	5	6	7	8	9	10
Volume (yds3)	8	8.5	9	9	9	9	9	9	9	9
Dump Number	11	12	13	14	15	16	17	18	19	20
Volume (yds3)	9	9	9	0	0					
Total Volume (yds3)	115.5									
Total Cycle Time	19.75	minutes								
TPH (based on density)=	521.20	TPH	Compared To [illegible]	528.2 [illegible]	TPH from weighed sample [illegible]					
			Difference [illegible]	1[illegible]						
DENSITY CALCUCATIONS										
		Pail Volume (ft3) =	0.48169	diameter =	0.8399	ft				
		Pail Weight =	25 lbs	height =	0.869	ft				
		weekly input		weekly input						
Conveyor Label	Description	Volumetric Pail Weight(lbs)	Tared Sample Weight (lbs)	Distance From Top If Not Full(cm)	Vol. To Use If Not Full	Density (lb/ft3)	Density Rank Should Be			
1	Pitrun	78	53	0	0.4817	110.029				
2	Sand	47	22	14.5	0.2181	100.863				
3	Screened Pit Run	79	54	0	0.4817	112.105				
4	Coarse Feed	73	48	0	0.4817	99.649				
5	Coarse Return	72	47	1.8	0.4490	104.684				
6	Fine Feed	78	53	0	0.4817	110.029				
7	Fine Return	74	49	0	0.4817	101.725				
8	Product	71	46	0	0.4817	95.497				

TPH CALCULATIONS		Trial #4 Thursday, October 10, 1996			Product :	ACR		Sample bigger		Factor
							weekly input		Weekly outpu	0.98018797
#1 - AGGREGATE SAMPLE								50 cm = 1.64 ft		ADJUSTED
Conveyor Label	Description	Length (ft) (axel to axle)	Axle Diam. (inches)	Length Per Revolution(ft)	Time Per Revolution(sec.)	# Of Belt Lengths/hr	Sample Pail Weight(lbs)	Mass Per 50cm Width	Tonnes Per Hour	Tonnes Per Hour
1	Pitrun	100.333	18	205.4	22.4	160.7	62	59.5	598.6	586.7
2	Sand	47.583	14	98.8	11.6	310.3	18	15.5	144.9	142.0
3	Screened Pit Run	25	12	53.1	10.2	352.9	71	68.5	391.6	383.8
4	Coarse Feed	39.5	10	81.6	19	189.5	48	45.5	214.5	210.2
5	Coarse Return	43	12	89.1	15.6	230.8	33	[illegible]	206.9	202.8
6	Fine Feed	47.5	12	98.1	19.2	187.5	45	[illegible]	252.4	247.4
7	Fine Return	40.75	12	84.6	22	163.6	54	[illegible]	228.0	223.5
8	Product	47.5	12	98.1	23.4	153.8	65	82.5	379.7	372.2
Production Rate According To Weigh Scales		Number Of Samples =		20			Average Weight From Scale:		372.15	
(at one minute intervals in TPH)										
1	2	3	4	5	6	7	8	9	10	
405	407	400	395	387	350	361	385	407	420	
11	12	13	14	15	16	17	18	19	20	
332	341	342	373	360	357	344	333	360	364	
AVERAGE TPH =	372.15	[illegible]		[illegible] from weighed sample						
#2 - LOADER ANALYSIS										
Bucket Capacity =	9	cubic yards								
Dump Number	1	2	3	4	5	6	7	8	9	10
Volume (yds3)	9	8.5	9	9	9.5	9	9.25	9	8.5	9.25
Dump Number	11	12	13	14	15	16	17	18	19	20
Volume (yds3)	9.5	9.5	9	9.25	0					
Total Volume (yds3)	127.25									
Total Cycle Time	20.5	minutes								
TPH (based on density)=	563.66	TPH	Comparative Difference [illegible]		[illegible] TPH from weighed sample					
DENSITY CALCUCATIONS										
		Pail Volume (ft3) =	0.48169	diameter =	0.8399	ft				
		Pail Weight =	25 lbs	height =	0.869	ft				
		weekly input		weekly input						
Conveyor Label	Description	Volumetric Pail Weight(lbs)	Tared Sample Weight (lbs)	Distance From Top If Not Full(cm)	Vol. To Use If Not Full	Density (lb/ft3)	Density Rank Should Be			
1	Pitrun	79	54	0	0.4817	112.105				
2	Sand	41	16	17.33	0.1666	96.029				
3	Screened Pit Run	83	58	0	0.4817	120.409				
4	Coarse Feed	71	46	2	0.4453	103.293				
5	Coarse Return	60	35	7.67	0.3423	102.240				
6	Fine Feed	69	44	0	0.4817	91.345				
7	Fine Return	75	50	0	0.4817	103.801				
8	Product	[illegible]	109	7.33	[illegible]	131.313				
		filled the bucket 2 times with all product								

TPH CALCULATIONS

Trial #6 Thursday, October 18, 1996

Product : Road Crush (20mm)

Note: Pitrun is now being collected from the feed hopper (not the same as previous)

#1 - AGGREGATE SAMPLE

							weekly input	50 cm = 1.64 ft	Weekly output
Conveyor Label	Description	Length (ft) (axel to axle)	Axle Diam. (inches)	Length Per Revolution(ft)	Time Per Revolution(sec.)	# Of Belt Lengths/hr	Sample Pail Weight(lbs)	Mass Per 50cm Width	Tonnes Per Hour
1	Pitrun	47	18	98.7	30.8	116.9	195	192.5	677.0

AVERAGE TPH = [illegible] 425 from Bill Laisse [illegible]

#2 - LOADER ANALYSIS

Dump Number	1	2	3	4	5	6	7	8	9	10
Volume (yds3)	9.5	9.5	9	9.5	9	9	9.5	9	9	9
Dump Number	11	12	13	14	15	16	17	18	19	20
Volume (yds3)	9.5	9	9	0	0					
Total Volume (yds3)	119.5									
Total Cycle Time	19.67	minutes								
TPH (based on density)=	562.31	TPH	Compared to [illegible] 677.0 [illegible] TPH from weighed sample % Difference [illegible]							

DENSITY CALCUCATIONS

Pail Volume (ft3) = 0.48169

Pail Weight = 25 lbs

diameter = 0.8399 ft

height = 0.869 ft

		weekly input		weekly input			
Conveyor Label	Description	Volumetric Pail Weight(lbs)	Tared Sample Weight (lbs)	Distance From Top If Not Full(cm)	Vol. To Use If Not Full	Density (lb/ft3)	Density Rank Should Be
1	Pitrun	82	57	0	0.4817	118.333	
2	Sand	66	41	1	0.4635	88.455	
8	Product	73	48	0	0.4817	99.649	

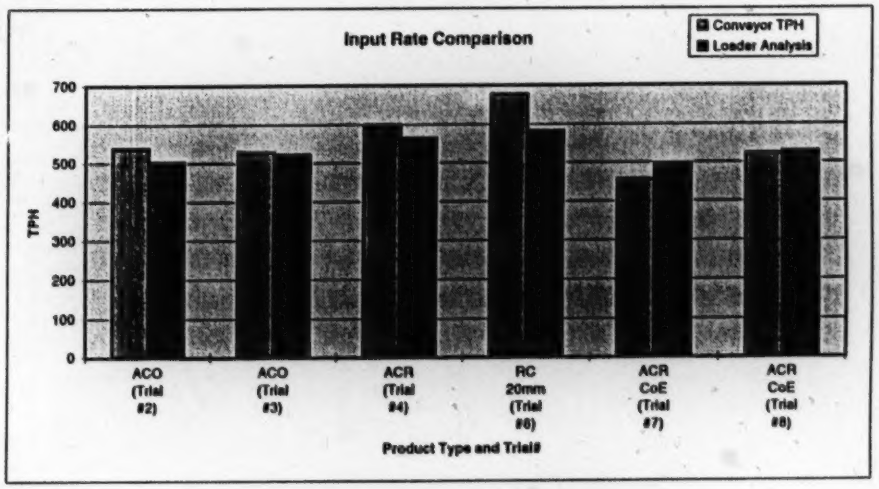
TPH CALCULATIONS		**Trial #7 Tuesday, October 22, 1996**			**Product :**	**ACR (City of Edmonton)**				
		Note: Pitrun is now being collected from the feed hopper (not the same as previous)					**weekly input**		**Weekly output**	
#1 - AGGREGATE SAMPLE								**50 cm = 1.64 ft**		
Conveyor Label	**Description**	**Length (ft) (axel to axle)**	**Axle Diam. (inches)**	**Length Per Revolution(ft)**	**Time Per Revolution(sec.)**	**# Of Belt Lengths/hr**	**Sample Pail Weight(lbs)**	**Mass Per 50cm Width**	**Tonnes Per Hour**	
1	Pitrun	47	18	98.7	30.8	116.9	132	129.5	455.4	
Production Rate According To Weigh Scales			**Number Of Samples =**	20			**Average Weight From Scale:**		321.15	
(at one minute intervals in TPH)										
1 330	2 325	3 335	4 331	5 341	6 347	7 323	8 319	9 315	10 317	
11 310	12 310	13 313	14 316	15 325	16 315	17 318	18 308	19 320	20 305	
AVERAGE TPH =	321.15	As Compared To [illegible] [illegible] % Difference [illegible]		TPH from weighed sample [illegible]						
#2 - LOADER ANALYSIS										
Dump Number	1	2	3	4	5	6	7	8	9	10
Volume (yds3)	8.5	9.5	9.5	9	9	9	9	9	9	9.5
Dump Number	11	12	13	14	15	16	17	18	19	20
Volume (yds3)	9	9	9	0	0					
Total Volume (yds3)	118									
Total Cycle Time	20.75	minutes								
TPH (based on density)=	494.47	TPH	Compared to [illegible] % Difference [illegible]		TPH from weighed sample [illegible]					
DENSITY CALCUCATIONS										
		Pail Volume (ft3) =	0.48169	**diameter =**	0.8399	ft				
		Pail Weight =	25 lbs	**height =**	0.869	ft				
		weekly input		**weekly input**						
Conveyor Label	**Description**	**Volumetric Pail Weight(lbs)**	**Tared Sample Weight (lbs)**	**Distance From Top If Not Full(cm)**	**Vol. To Use If Not Full**	**Density (lb/ft3)**	**Density Rank Should Be**			
1	Pitrun	49	24	14.2	0.2236	107.348				
2	Sand	64	39	2.5	0.4362	89.399				
8	Product	78	53	0	0.4817	110.029				

TPH CALCULATIONS		**Trial #8 Thursday, October 24, 1996**			**Product :**	**ACR (City of Edmonton)**				
		Note: Pitrun is now being collected from the feed hopper (not the same as previous)					**weekly input**	**50 cm = 1.64 ft**	**Weekly output**	
#1 - AGGREGATE SAMPLE										
Conveyor Label	**Description**	**Length (ft) (axel to axle)**	**Axle Diam. (inches)**	**Length Per Revolution(ft)**	**Time Per Revolution(sec.)**	**# Of Belt Lengths/hr**	**Sample Pail Weight(lbs)**	**Mass Per 50cm Width**	**Tonnes Per Hour**	
1	Pitrun	47	18	98.7	30.8	116.9	151	148.5	522.2	
							Average Weight From Scale:		362.75	
Production Rate According To Weigh Scales			**Number Of Samples =**	20						
(at one minute intervals in TPH)										
1	2	3	4	5	6	7	8	9	10	
367	354	374	380	377	331	348	375	371	361	
11	12	13	14	15	16	17	18	19	20	
320	300	367	374	373	365	375	345	405	363	
AVERAGE TPH =	362.75	% Compared To [illegible]		TPH from weighed sample						
#2 - LOADER ANALYSIS										
Dump Number	1	2	3	4	5	6	7	8	9	10
Volume (yds3)	9	9	9	9	9	9	9	9	9	9
Dump Number	11	12	13	14	15	16	17	18	19	20
Volume (yds3)	9	9	9	9	0					
Total Volume (yds3)	126									
Total Cycle Time	20.85	**minutes**								
TPH (based on density)=	528.43	**TPH**	Compared To [illegible] % Difference	[illegible]	TPH from weighed sample [illegible]					
DENSITY CALCUCATIONS										
		Pail Volume (ft3) =	0.48169	**diameter =**	0.8399	ft				
		Pail Weight =	25 lbs	**height =**	0.869	ft				
		weekly input		**weekly input**						
Conveyor Label	**Description**	**Volumetric Pail Weight(lbs)**	**Tared Sample Weight (lbs)**	**Distance From Top If Not Full(cm)**	**Vol. To Use If Not Full**	**Density (lb/ft3)**	**Density Rank Should Be**			
1	Pitrun	77	52	0	0.4817	107.953				
2	Sand	58	33	5.33	0.3848	85.758				
8	Product	44	19	15.5	0.1999	95.028				

Comparisons Between The Conveyor TPH Calculations and The Loader Analysis / Weigh Scale Recordings

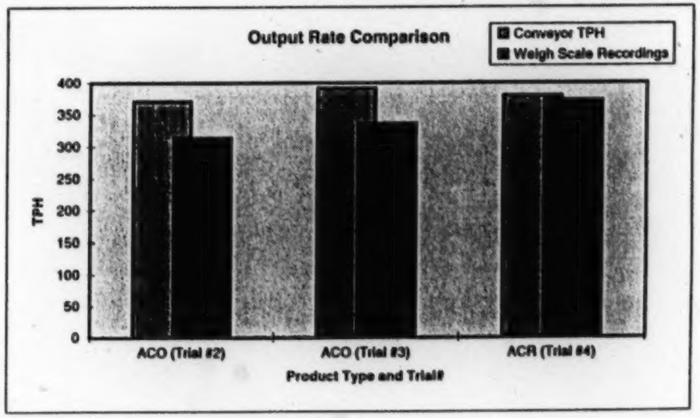
	Conveyor TPH	Loader Analysis	% Difference
ACO (Trial #2)	538.2	502.97	7.0
ACO (Trial #3)	528.2	521.2	1.3
ACR (Trial #4)	598.6	563.66	6.2
RC 20mm (Trial	677	582.3	16.3
ACR CoE (Trial	455.4	494.5	-7.9
ACR CoE (Trial	522.2	528.4	-1.2

Average Difference	6.6%

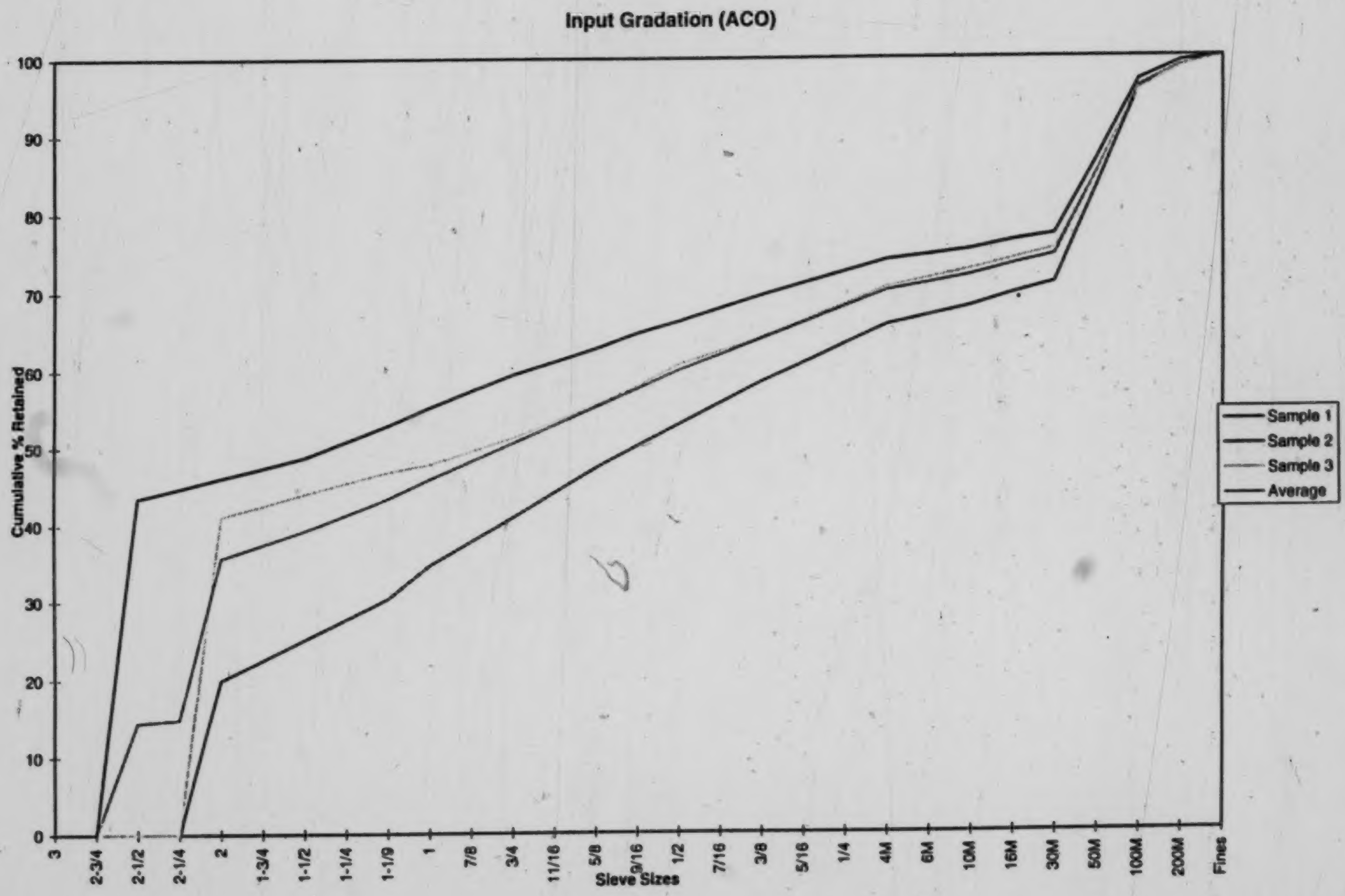


	Conveyor TPH	Weigh Scale Recordings	% Difference
ACO (Trial #2)	370.5	313.8	18.1
ACO (Trial #3)	391.2	335.5	16.6
ACR (Trial #4)	379.7	372.15	2.0

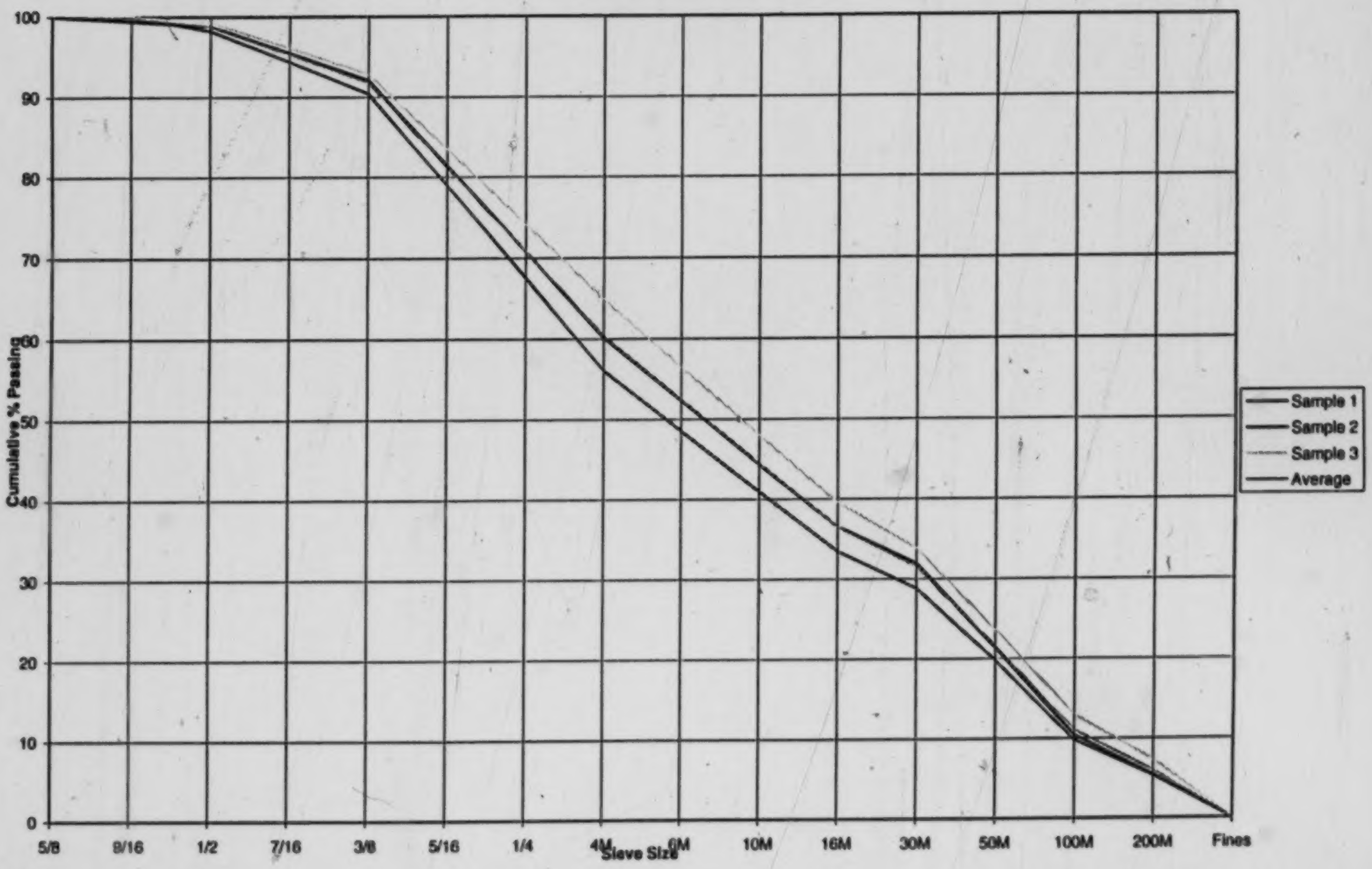
Average Difference	12.2%

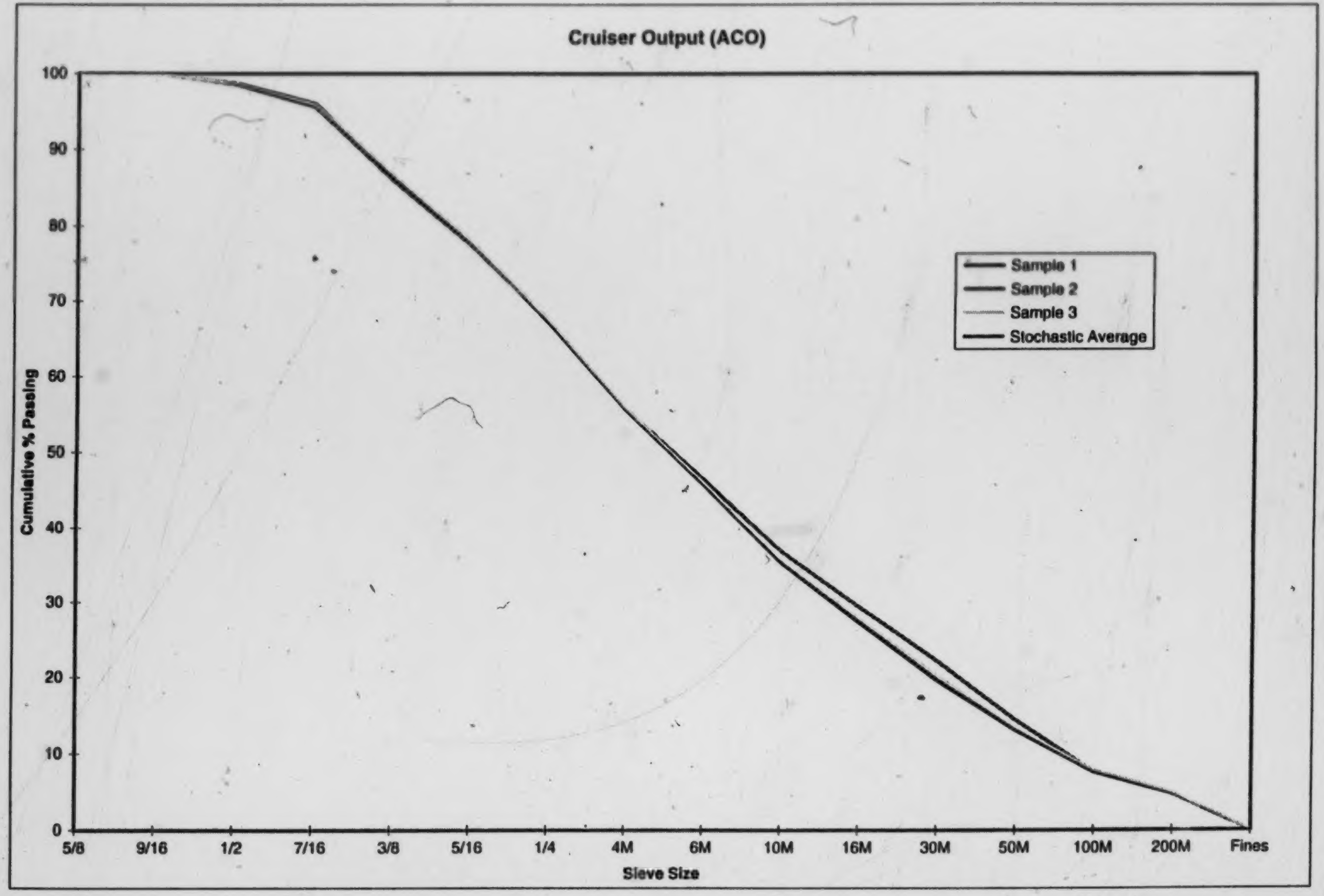


APPENDIX E

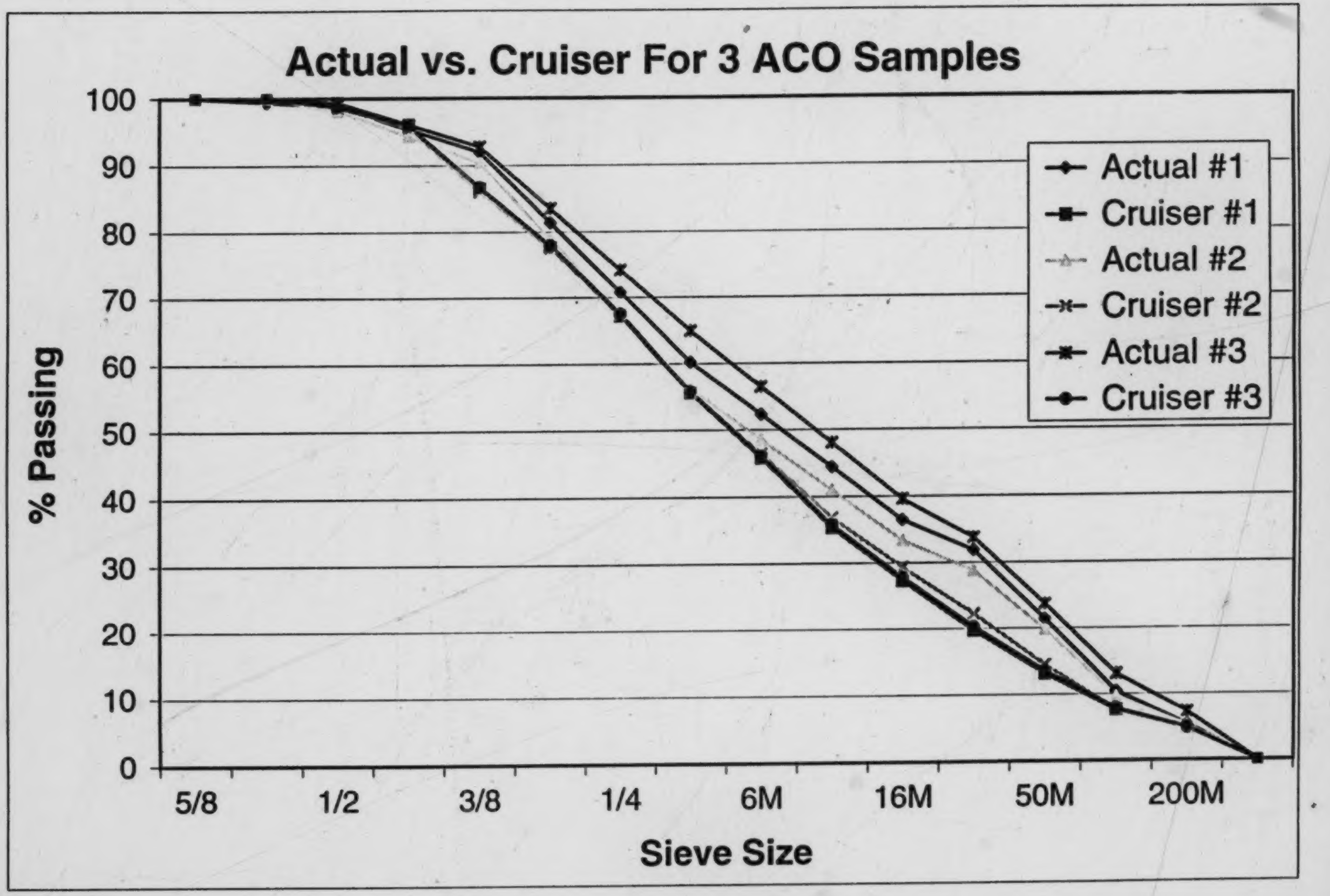

Input Gradation (ACO)
Cumulative % Retained
100
90
80
70
60
50
40
30
20
10
0
3
2-3/4
2-1/2
2-1/4
2
1-3/4
1-1/2
1-1/4
1-1/9
1
7/8
3/4
11/16
5/8
9/16
1/2
7/16
3/8
5/16
1/4
4M
6M
10M
16M
30M
50M
100M
200M
Fines
Sieve Sizes
Sample 1
Sample 2
Sample 3
Average

Output Gradation (ACO)

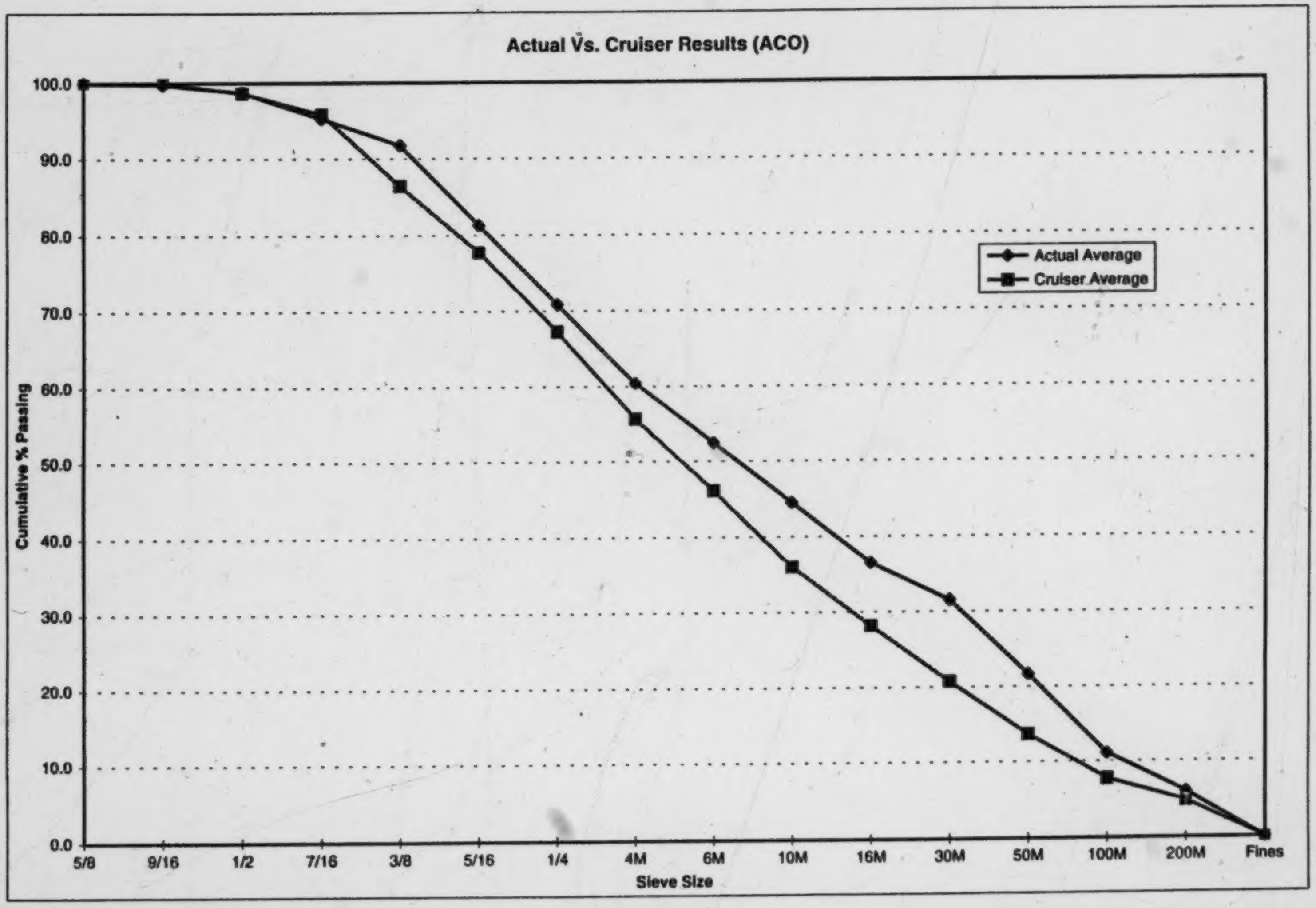




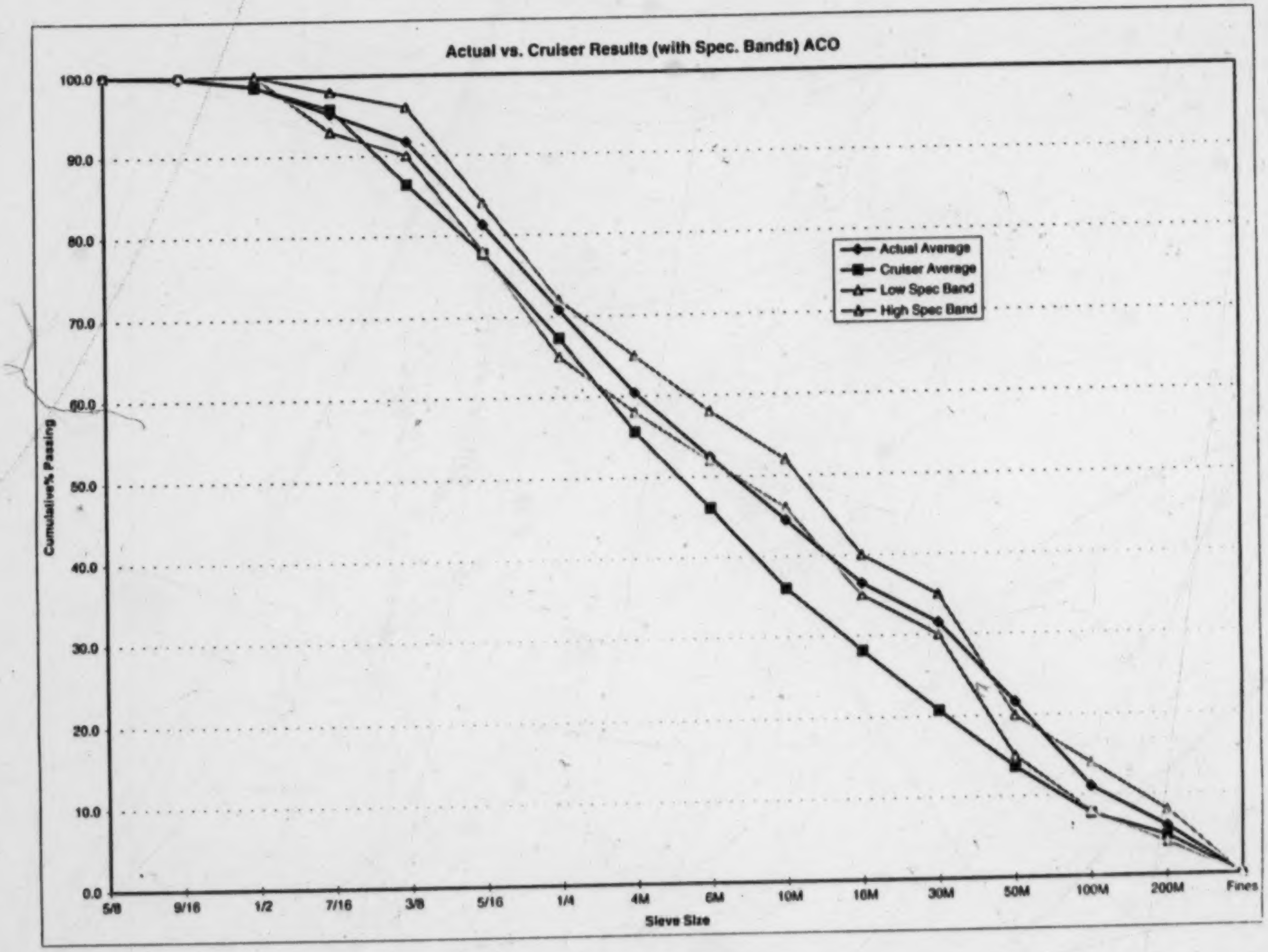
Cruiser Output (ACO)



Actual vs. Cruiser For 3 ACO Samples

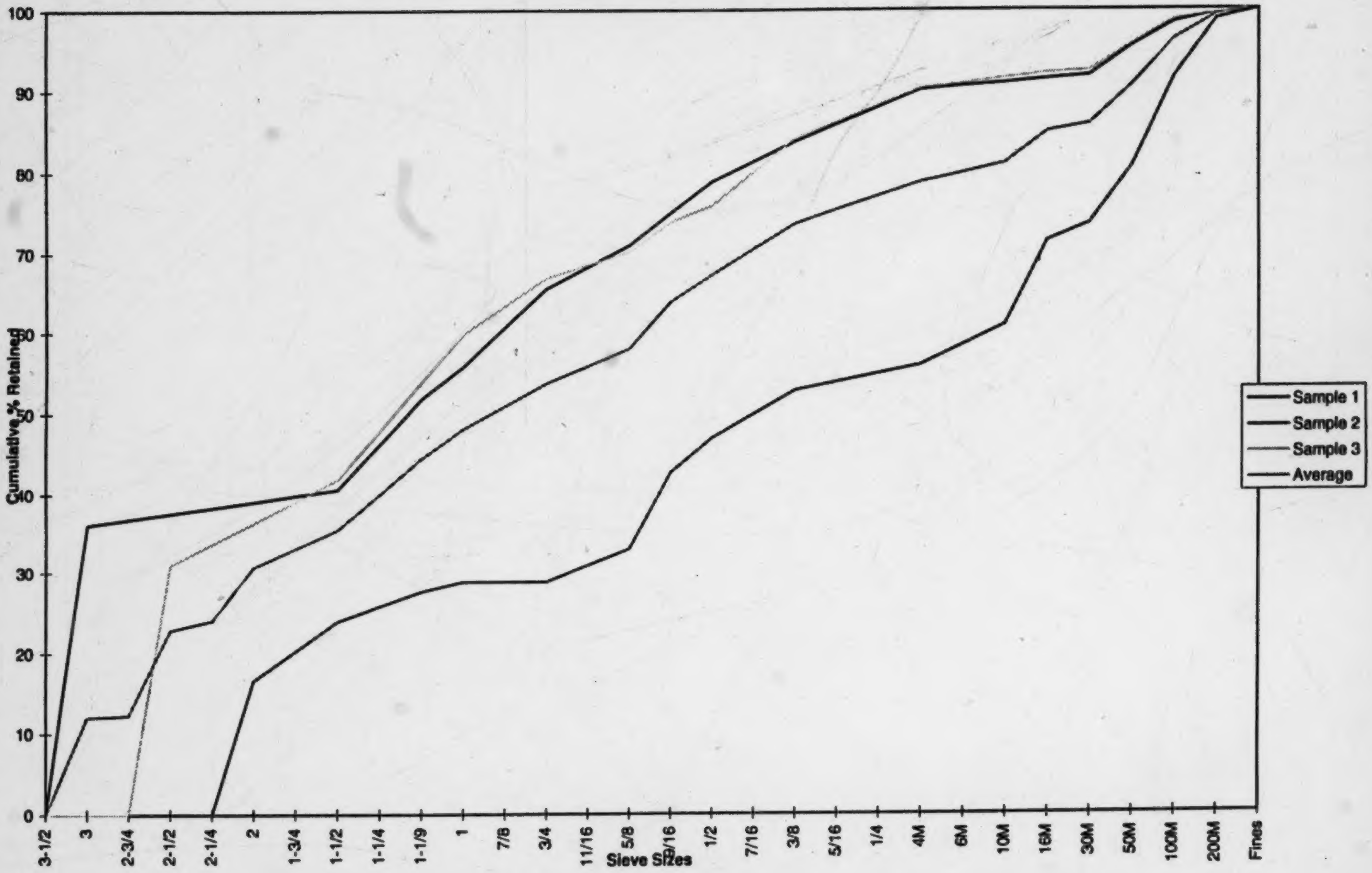


Actual Vs. Cruiser Results (ACO)

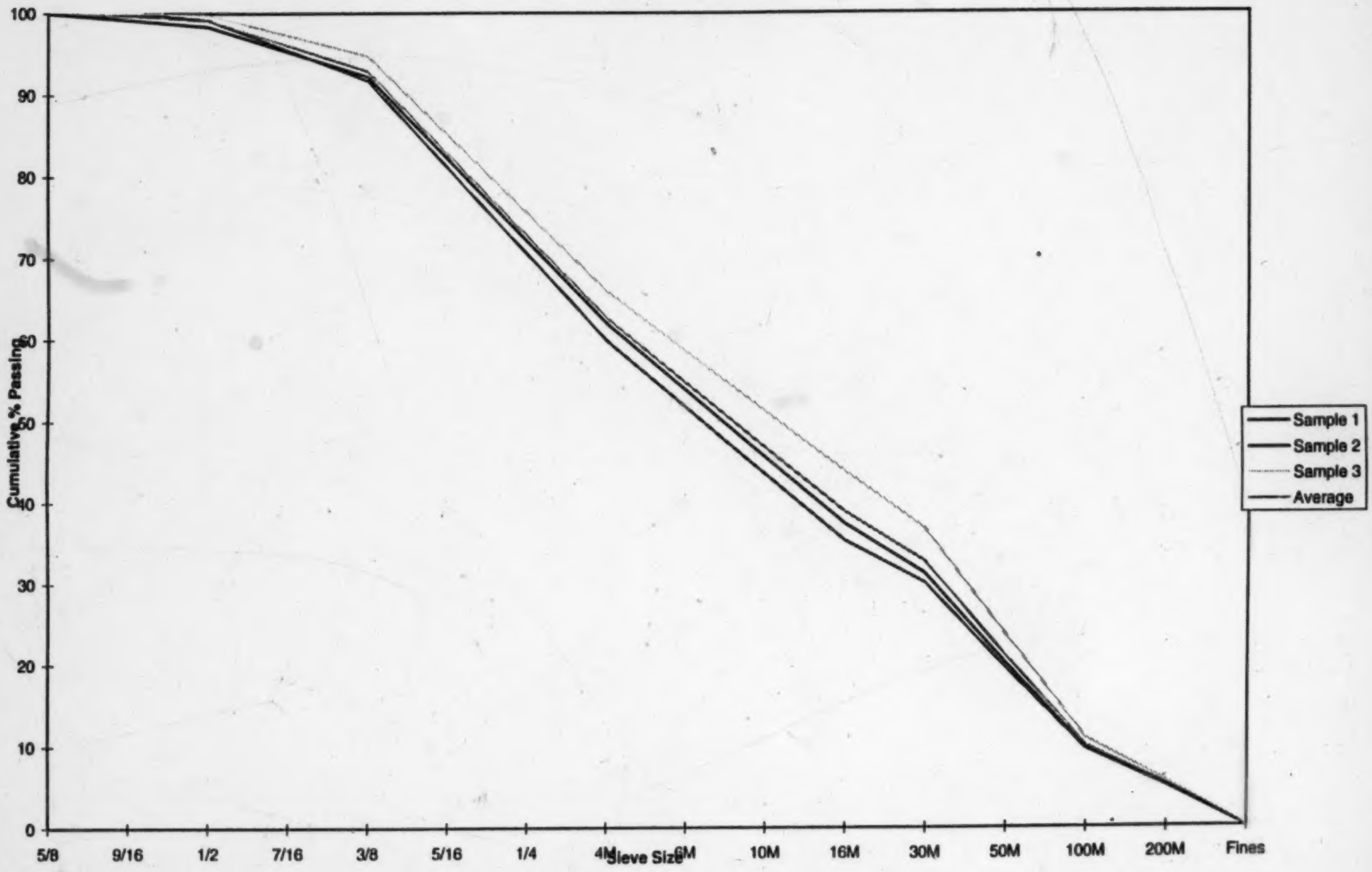

Actual vs. Cruiser Results (with Spec. Bands) ACO
Actual Average
Cruiser Average
Low Spec Band
High Spec Band
Cumulative% Passing
100.0
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
5/8
9/16
1/2
7/16
3/8
5/16
1/4
4M
6M
10M
16M
30M
50M
100M
200M
Fines
Sieve Size

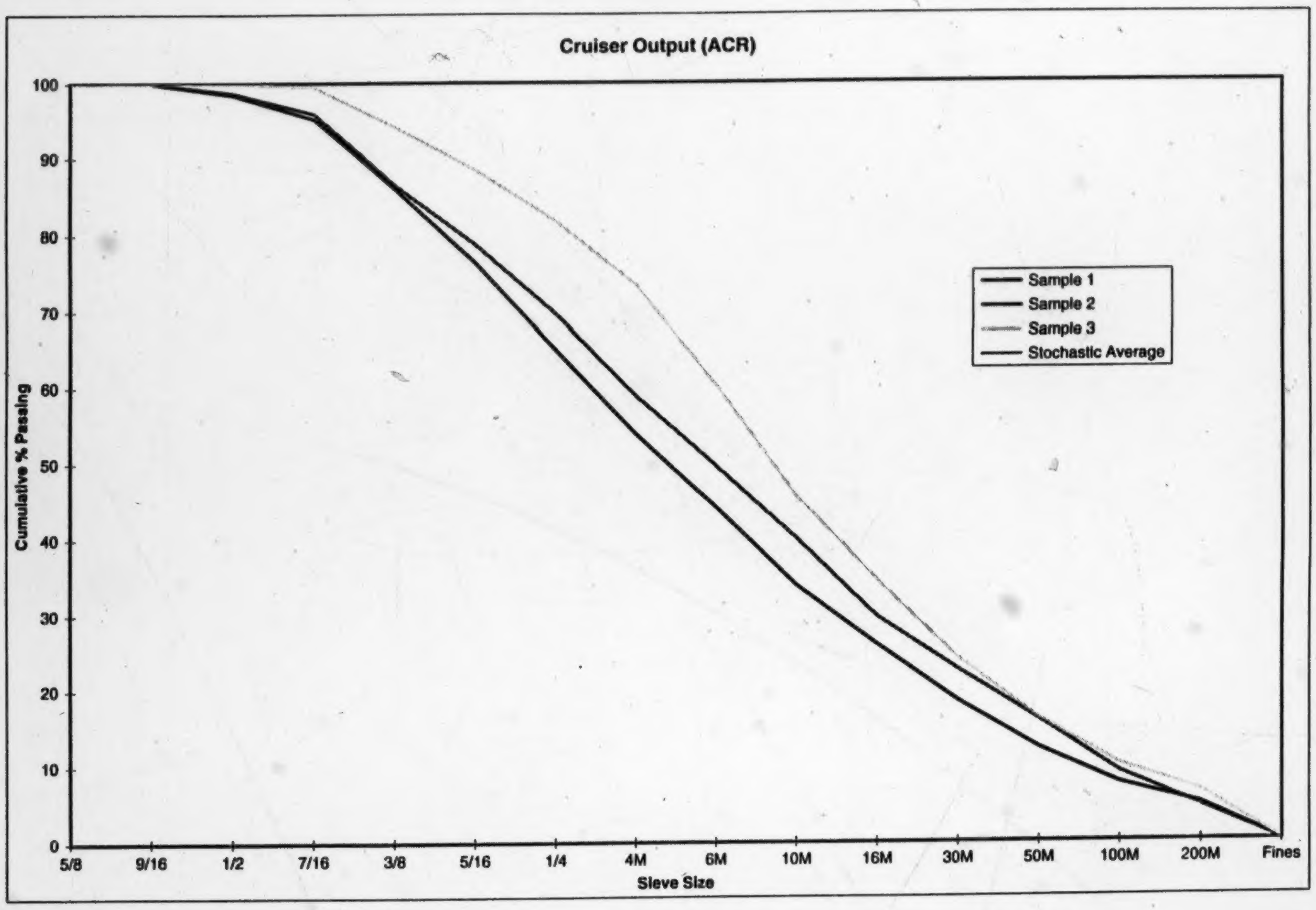
APPENDIX F

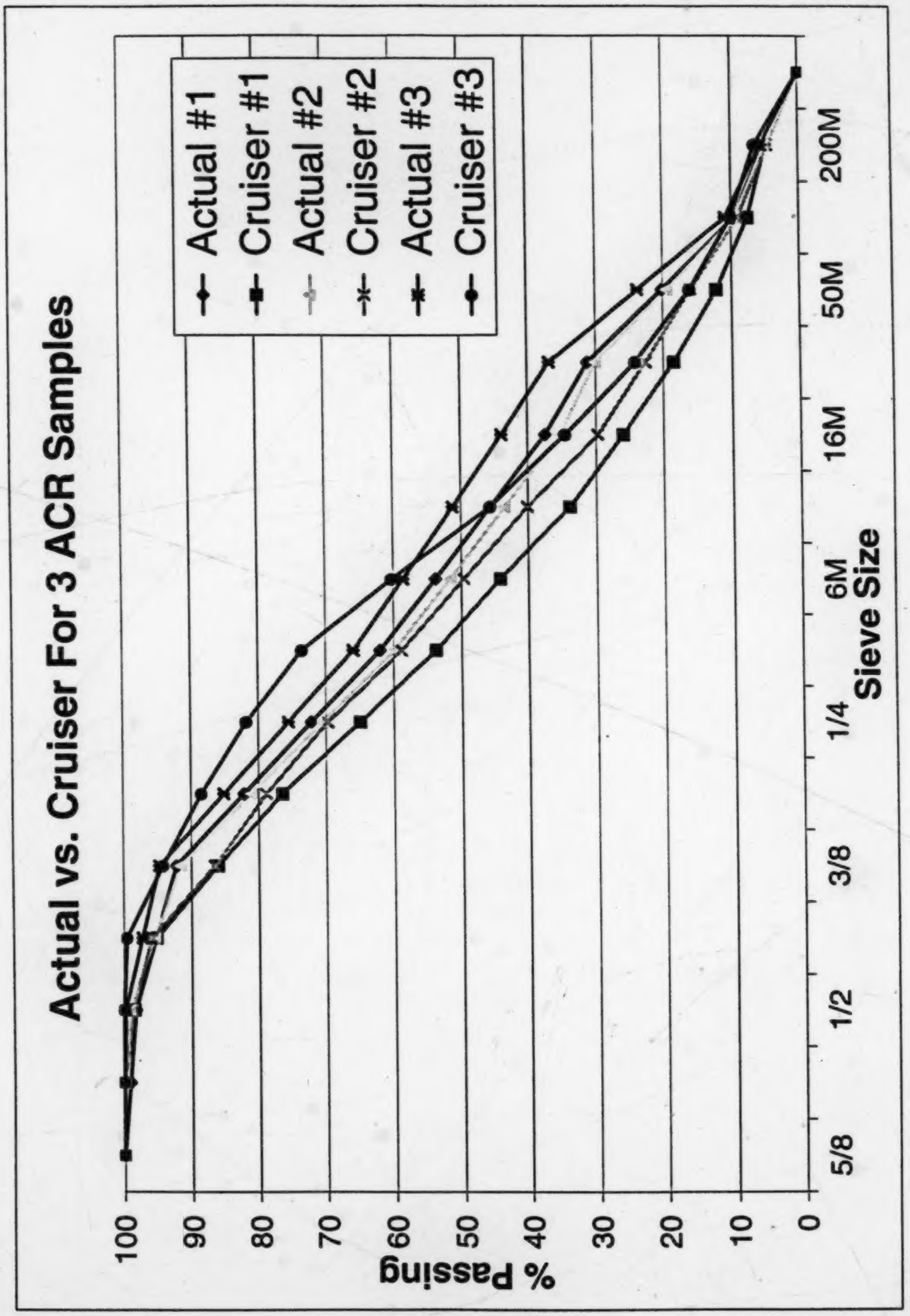
Input Gradation (ACR)

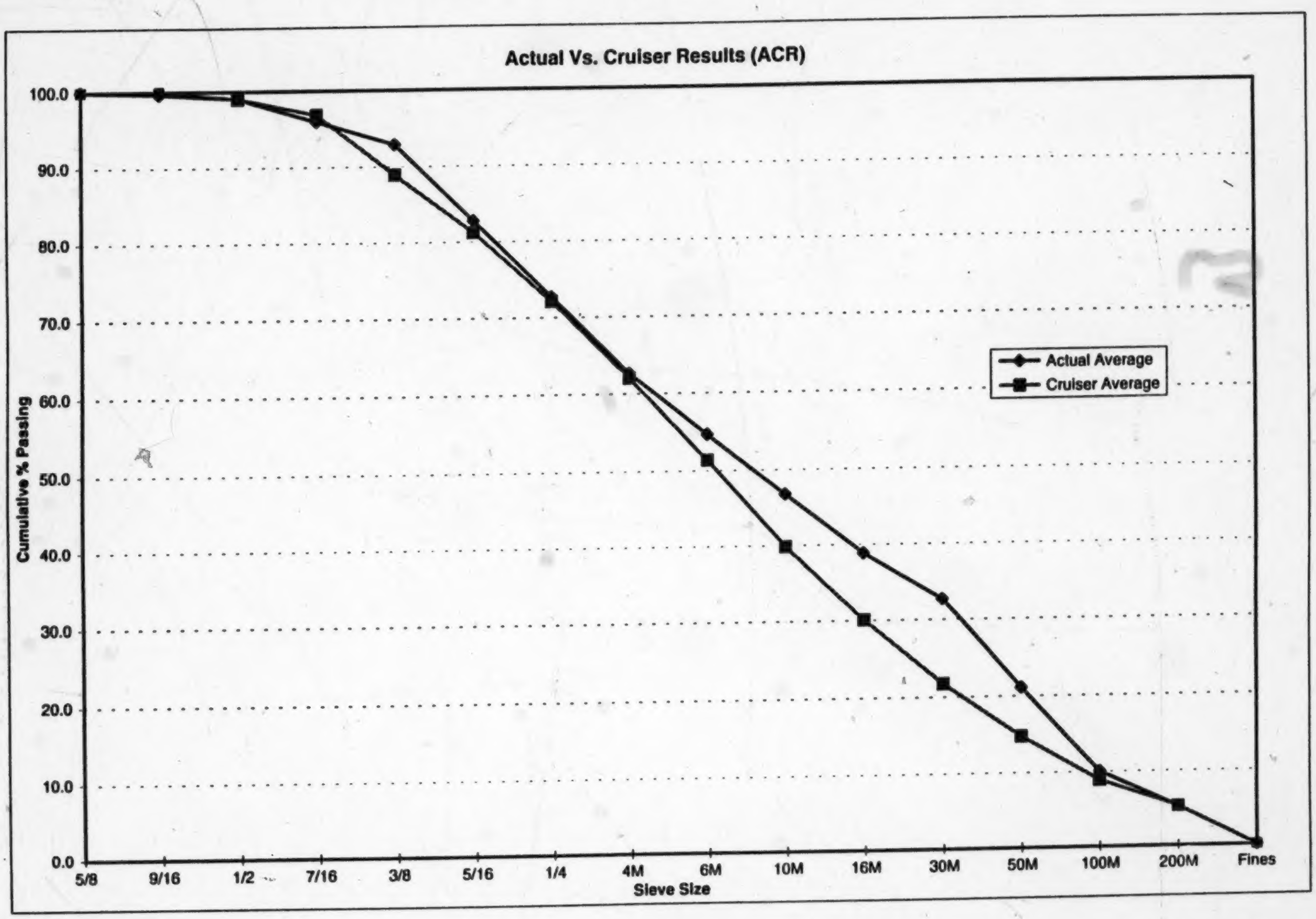


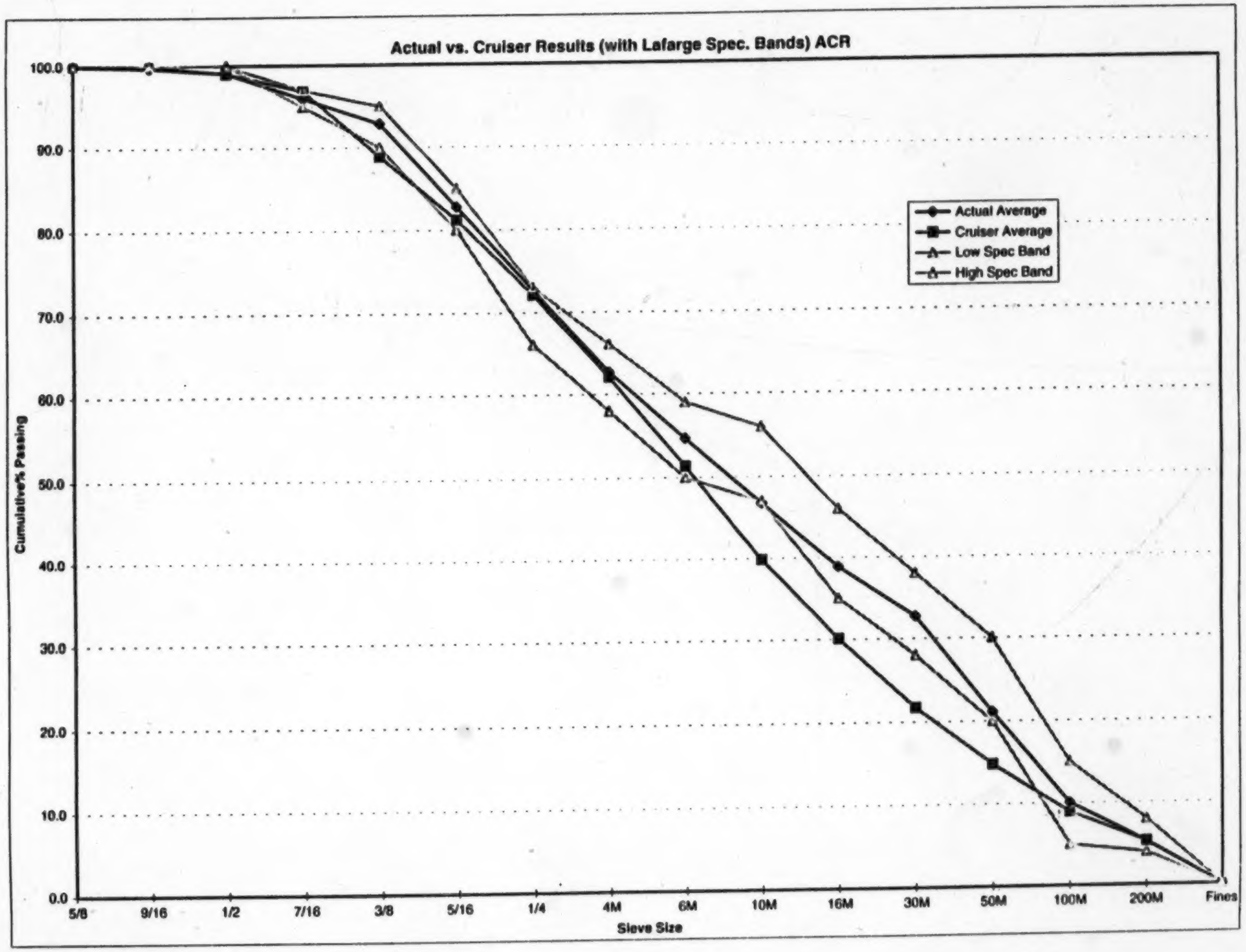
Output Gradation (ACR)

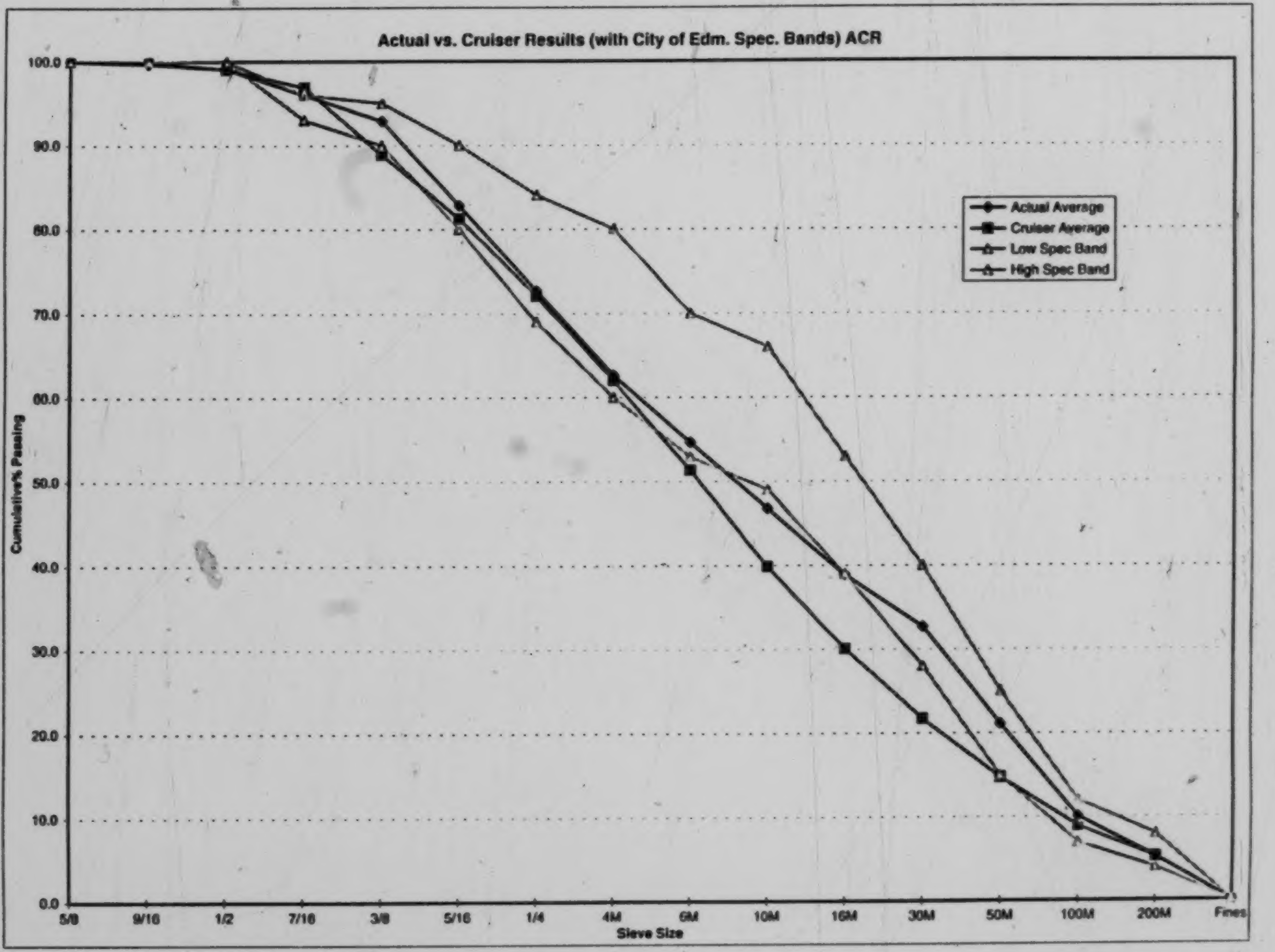



Cruiser Output (ACR)
Cumulative % Passing
100
90
80
70
60
50
40
30
20
10
0
5/8
9/16
1/2
7/16
3/8
5/16
1/4
4M
6M
10M
16M
30M
50M
100M
200M
Fines
Sieve Size
Sample 1
Sample 2
Sample 3
Stochastic Average


Actual vs. Cruiser For 3 ACR Samples
Actual #1
Cruiser #1
Actual #2
Cruiser #2
Actual #3
Cruiser #3
% Passing
100
90
80
70
60
50
40
30
20
10
0
Sieve Size
5/8
1/2
3/8
1/4
6M
16M
50M
200M


Actual Vs. Cruiser Results (ACR)
Cumulative % Passing
100.0
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
5/8
9/16
1/2
7/16
3/8
5/16
1/4
4M
6M
10M
16M
30M
50M
100M
200M
Fines
Sieve Size
Actual Average
Cruiser Average

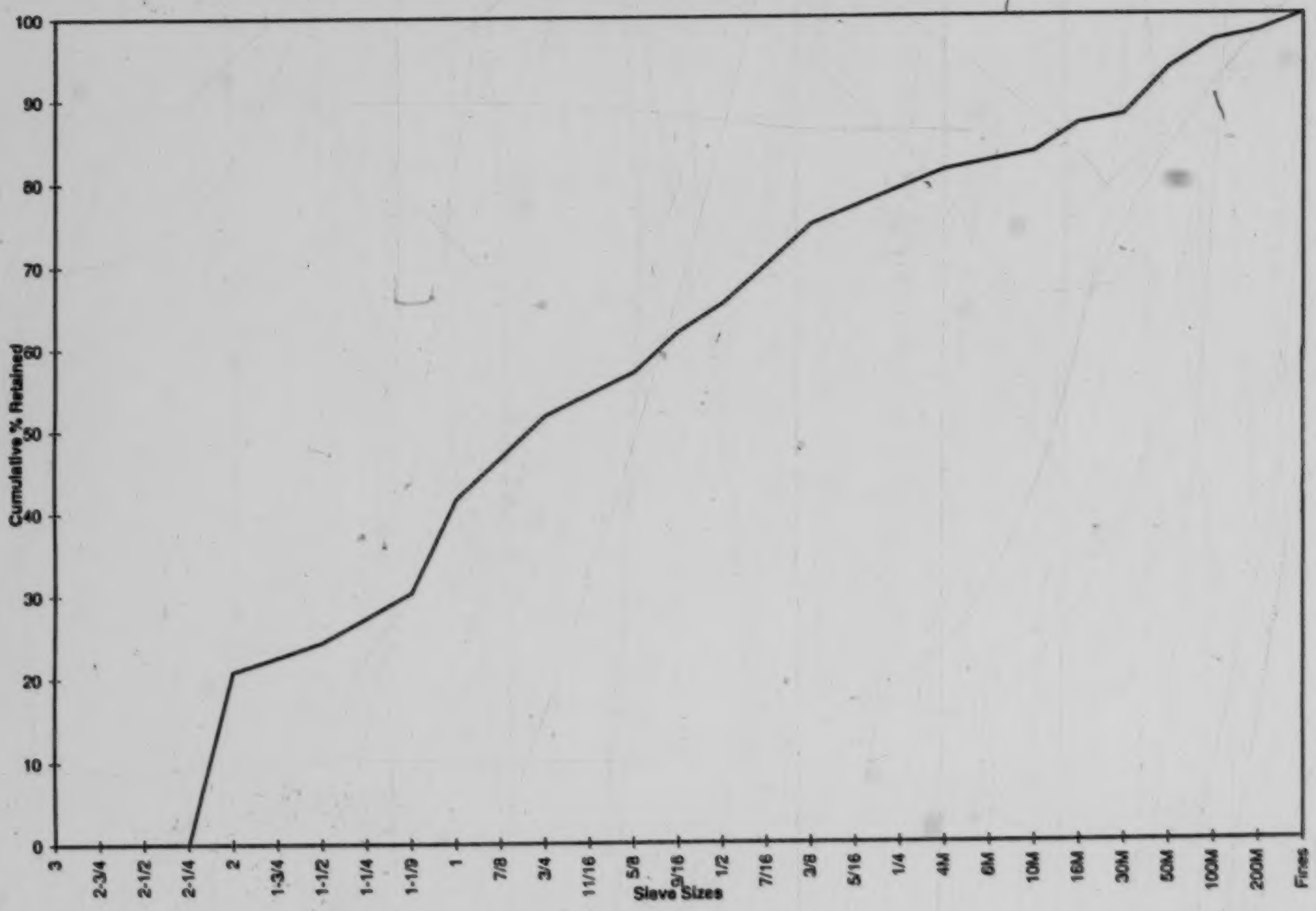

Actual vs. Cruiser Results (with Lafarge Spec. Bands) ACR
Actual Average
Cruiser Average
Low Spec Band
High Spec Band
Cumulative% Passing
100.0
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
5/8
9/16
1/2
7/16
3/8
5/16
1/4
4M
6M
10M
16M
30M
50M
100M
200M
Fines
Sieve Size

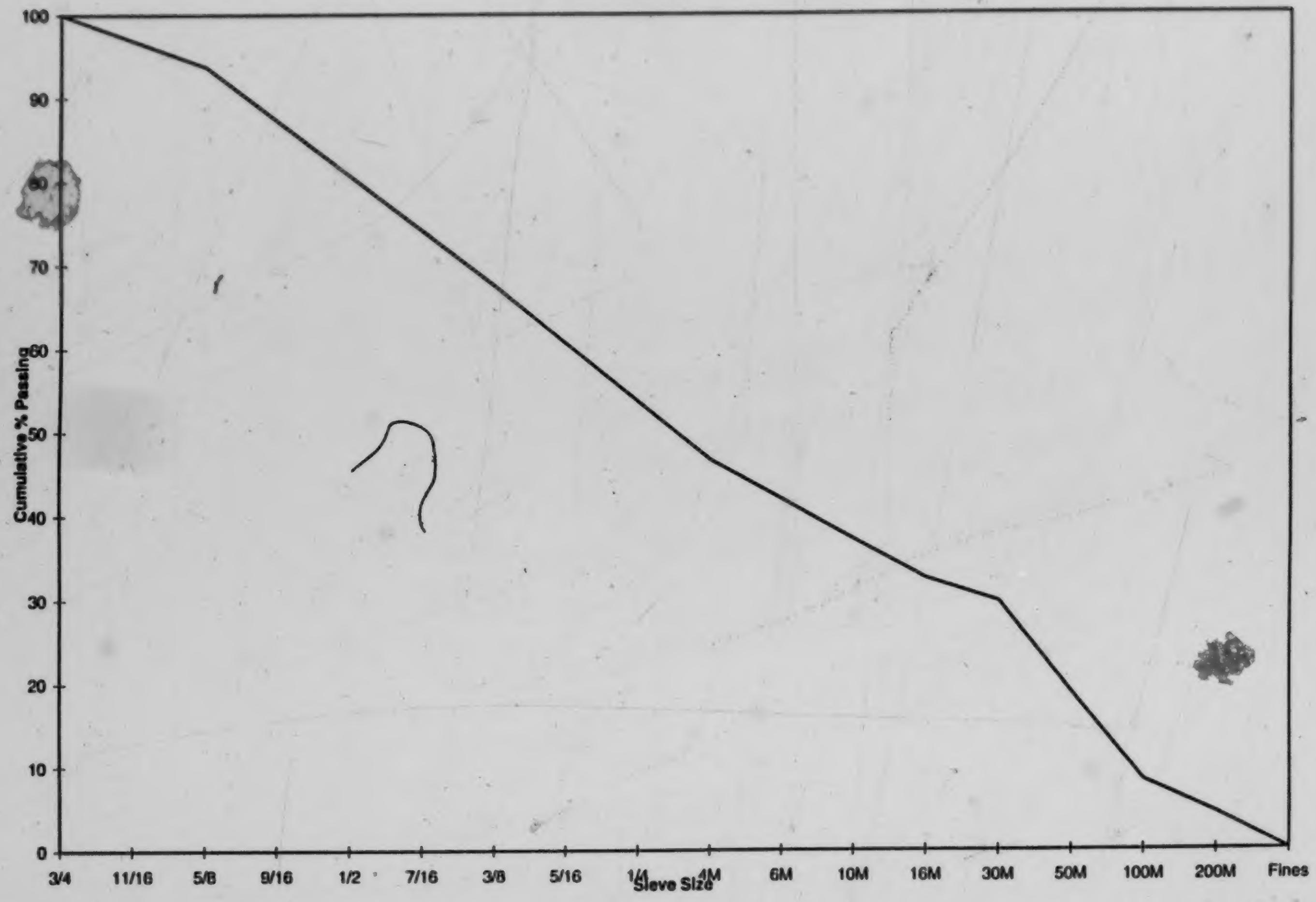


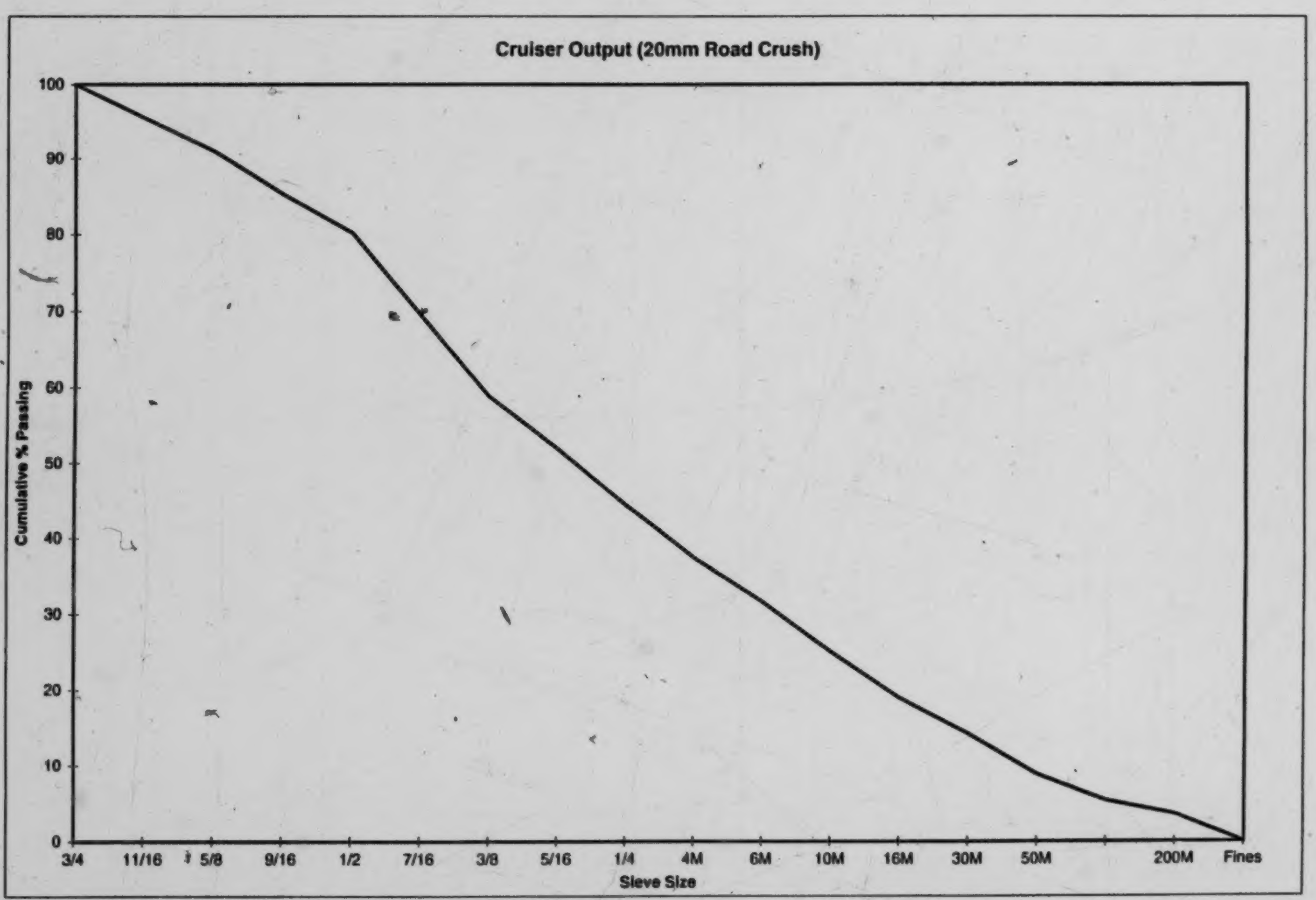
Actual vs. Cruiser Results (with City of Edm. Spec. Bands) ACR

APPENDIX G

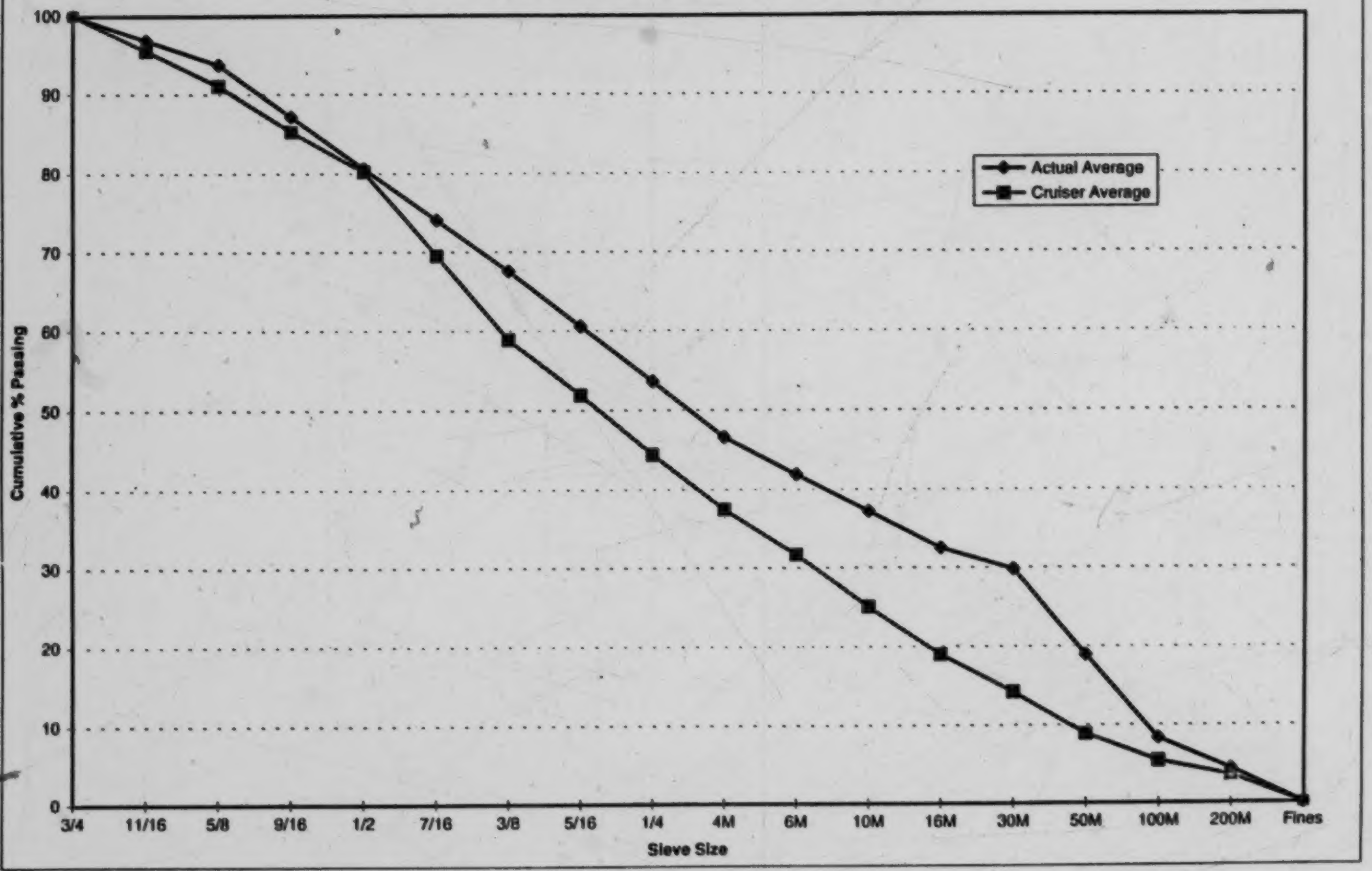
Input Gradation (20mm Road Crush)



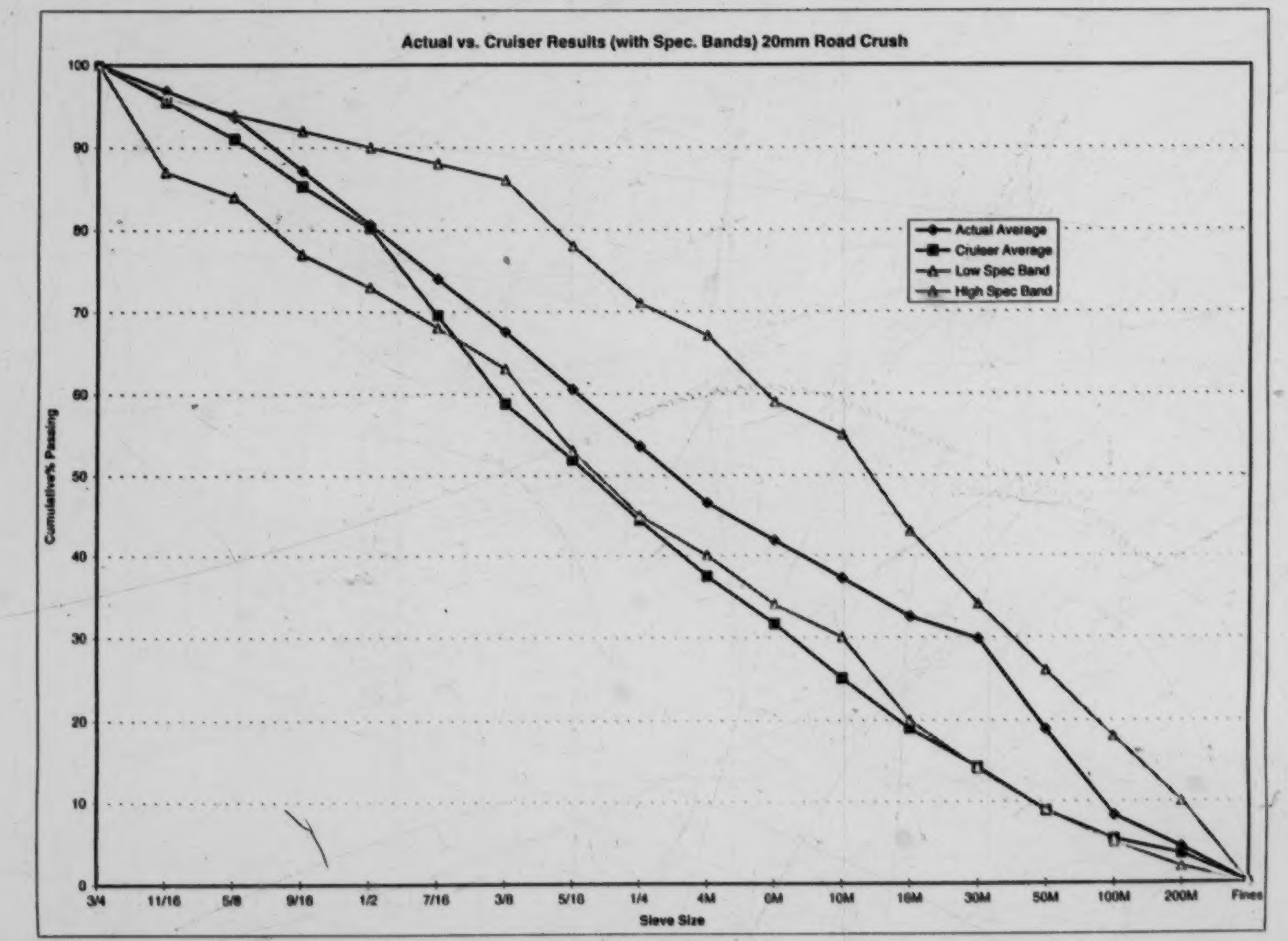

Output Gradation (20mm Road Crush)



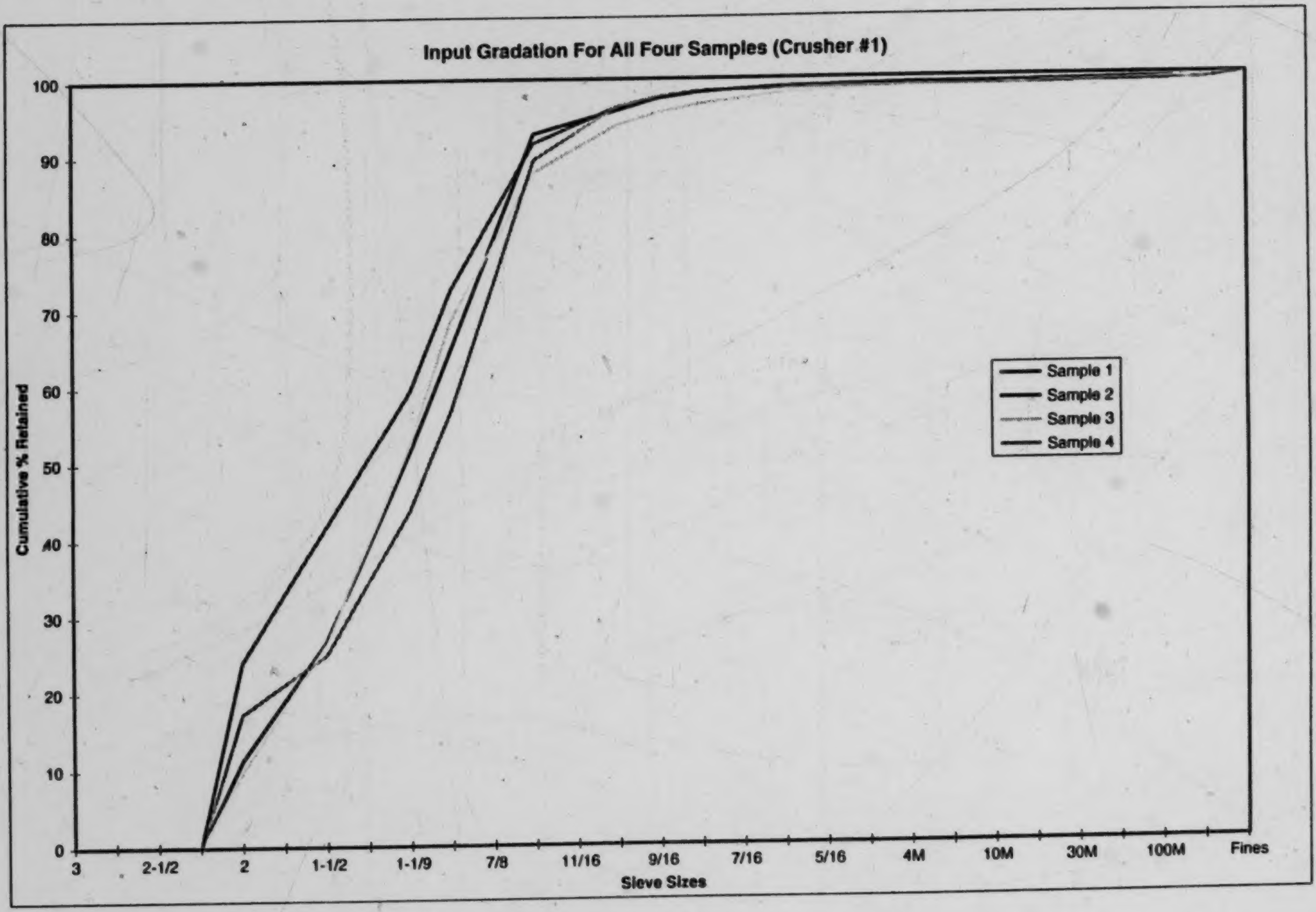
Cruiser Output (20mm Road Crush)

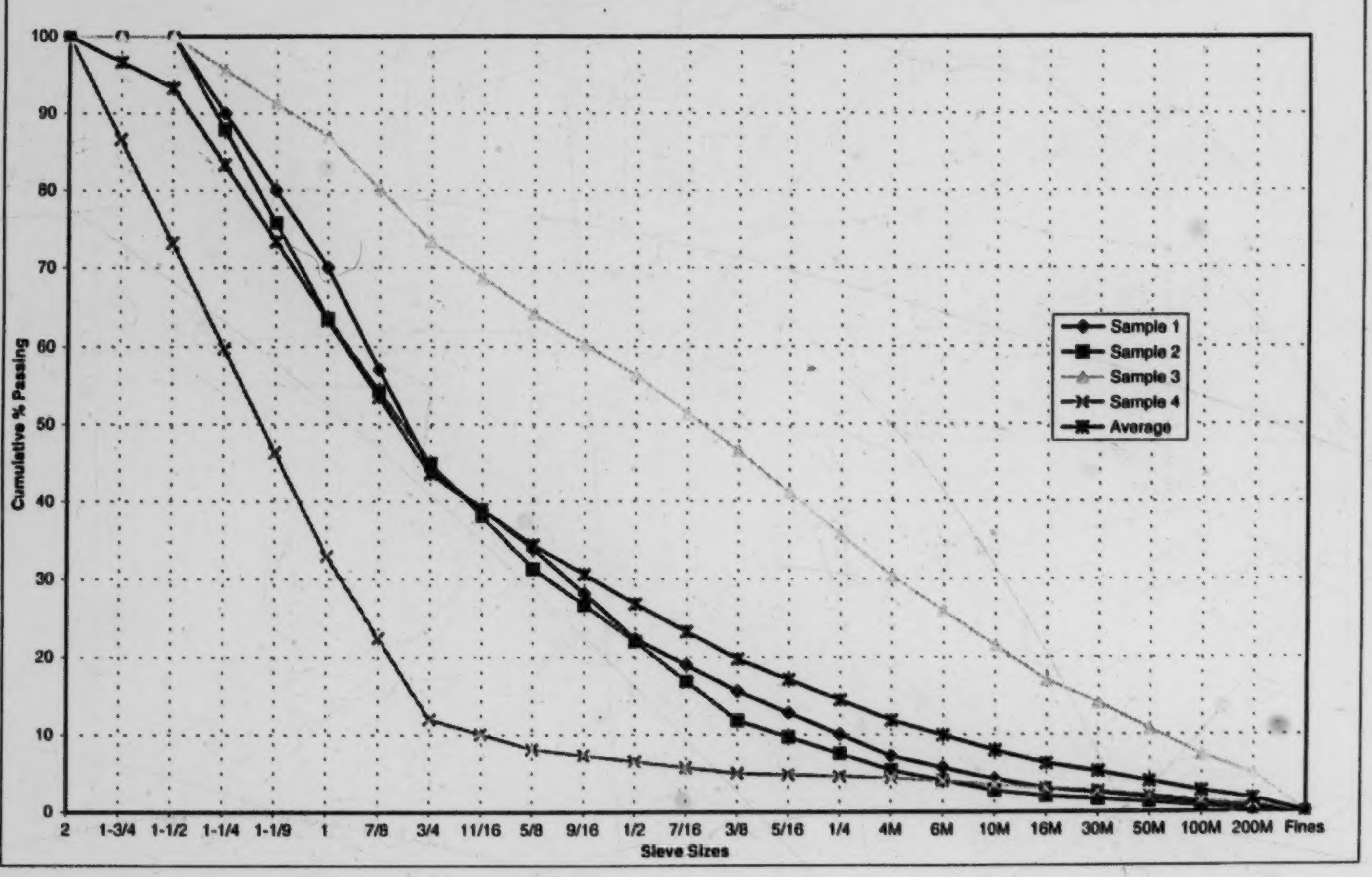


Actual Vs. Cruiser Results (20mm Road Crush)


Actual vs. Cruiser Results (with Spec. Bands) 20mm Road Crush
Actual Average
Cruiser Average
Low Spec Band
High Spec Band
Cumulative% Passing
0
10
20
30
40
50
60
70
80
90
100
3/4
11/16
5/8
9/16
1/2
7/16
3/8
5/16
1/4
4M
8M
10M
16M
30M
50M
100M
200M
Fines
Sieve Size

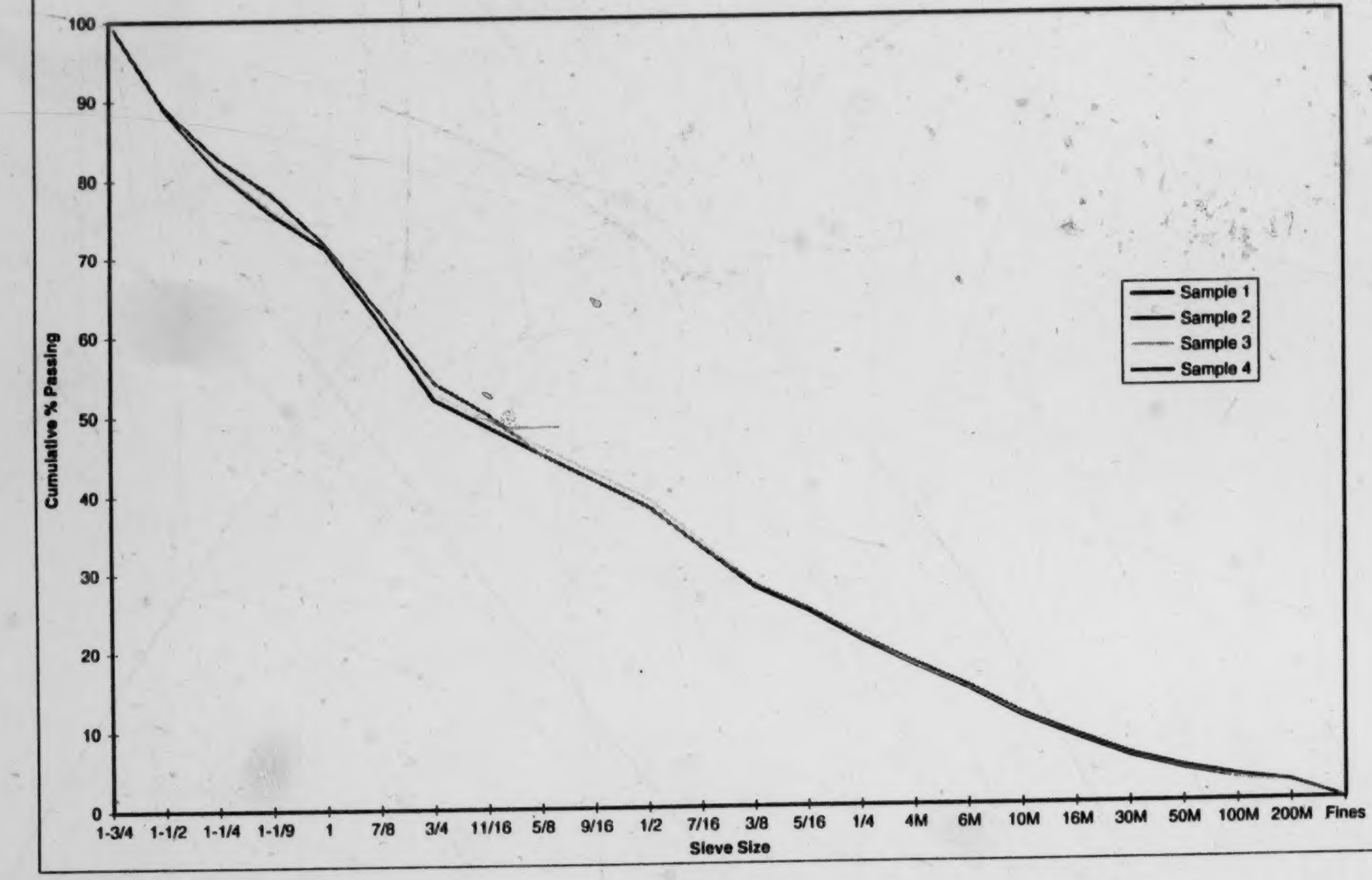
APPENDIX H

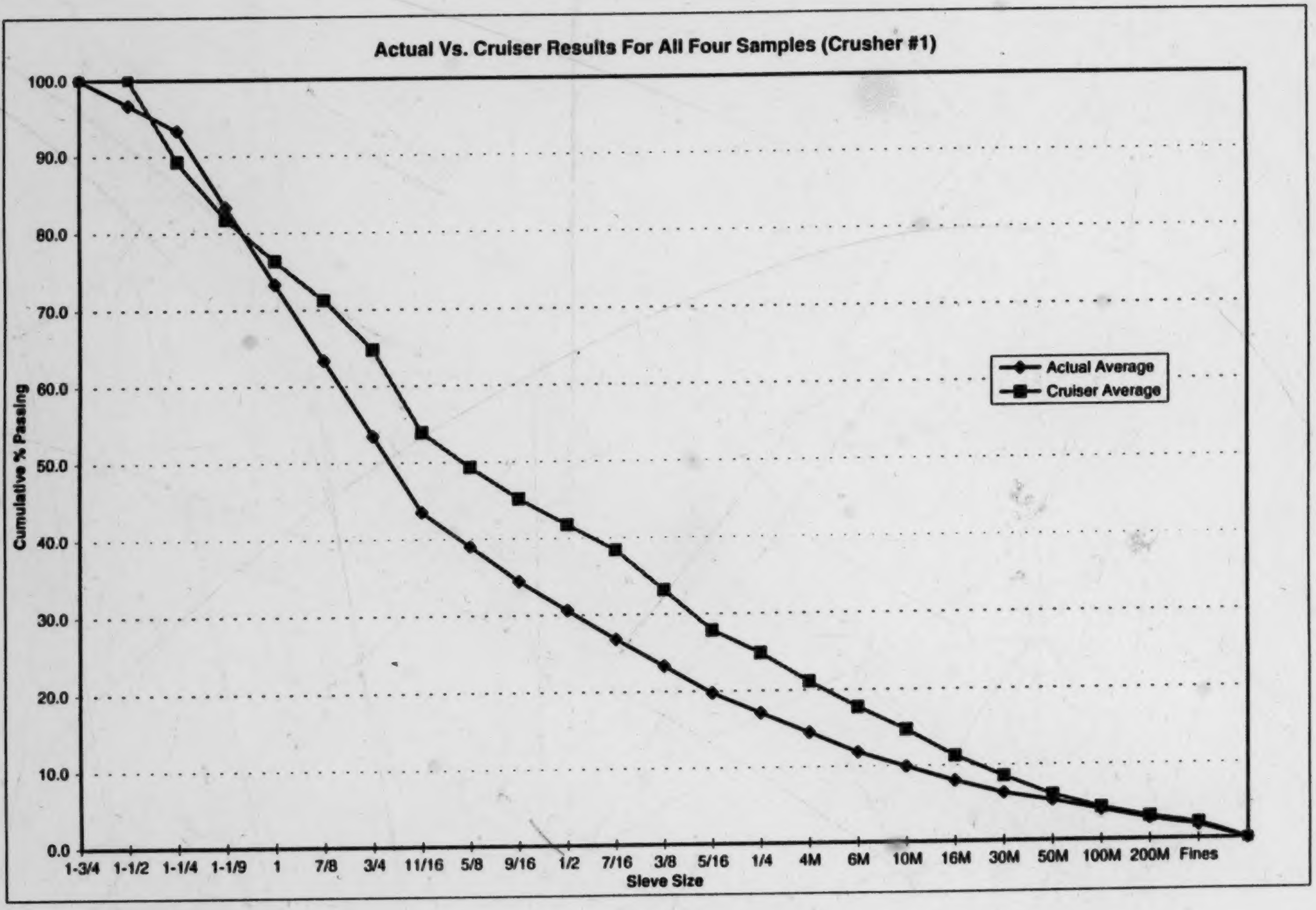

Input Gradation For All Four Samples (Crusher #1)



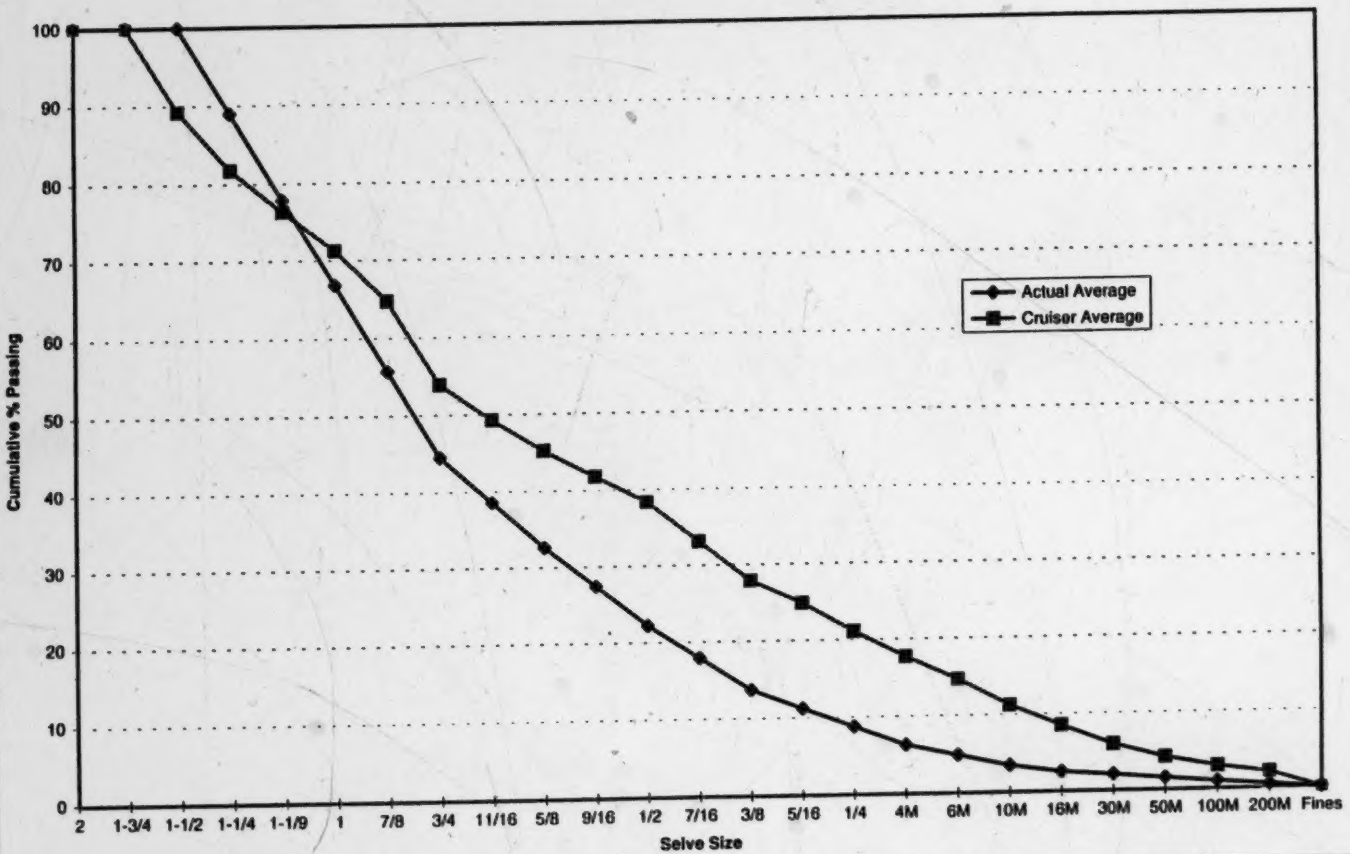
Output Gradation For All Four Samples (Crusher #1)

Cruiser Output For All Four Samples (Crusher #1)

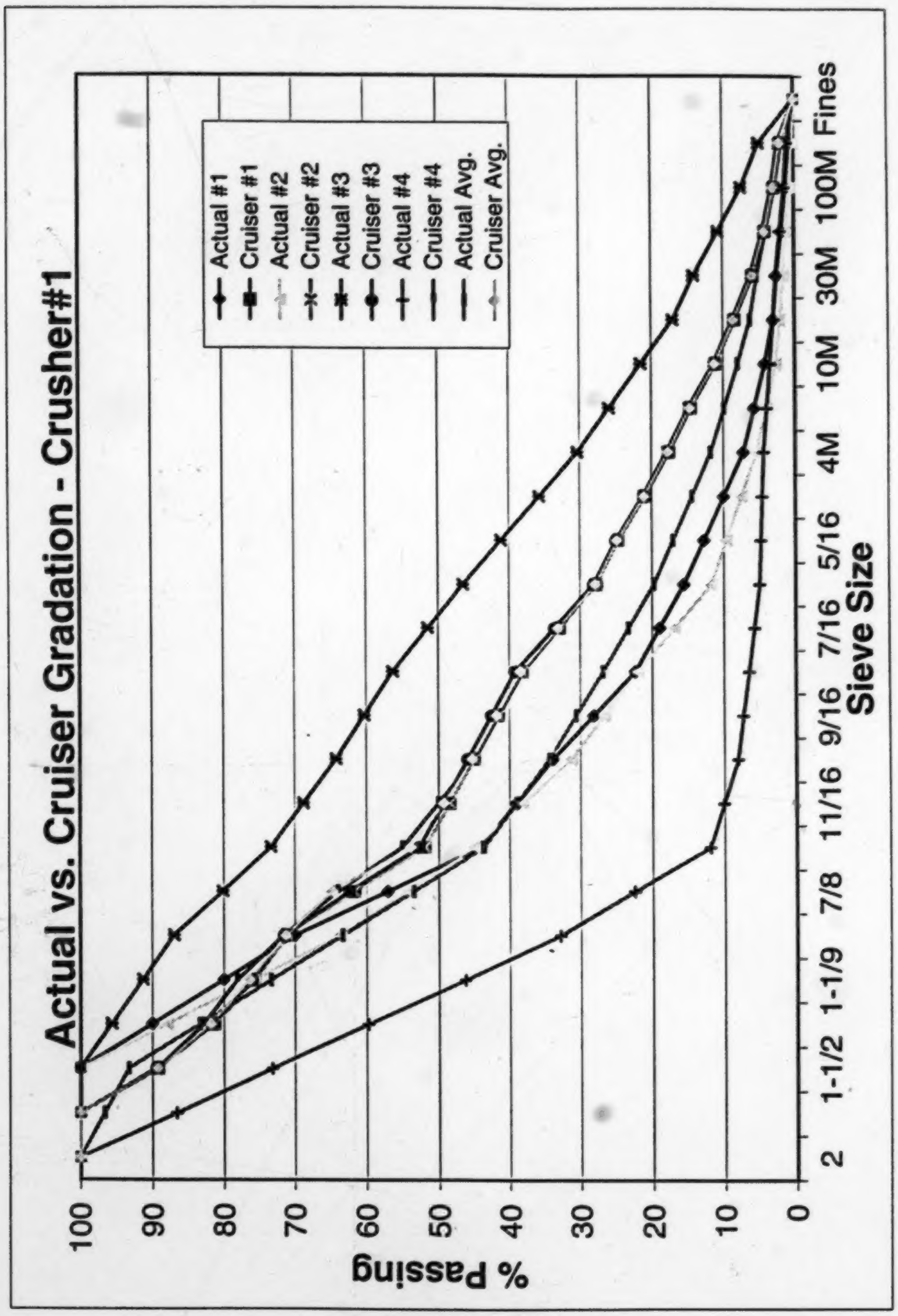


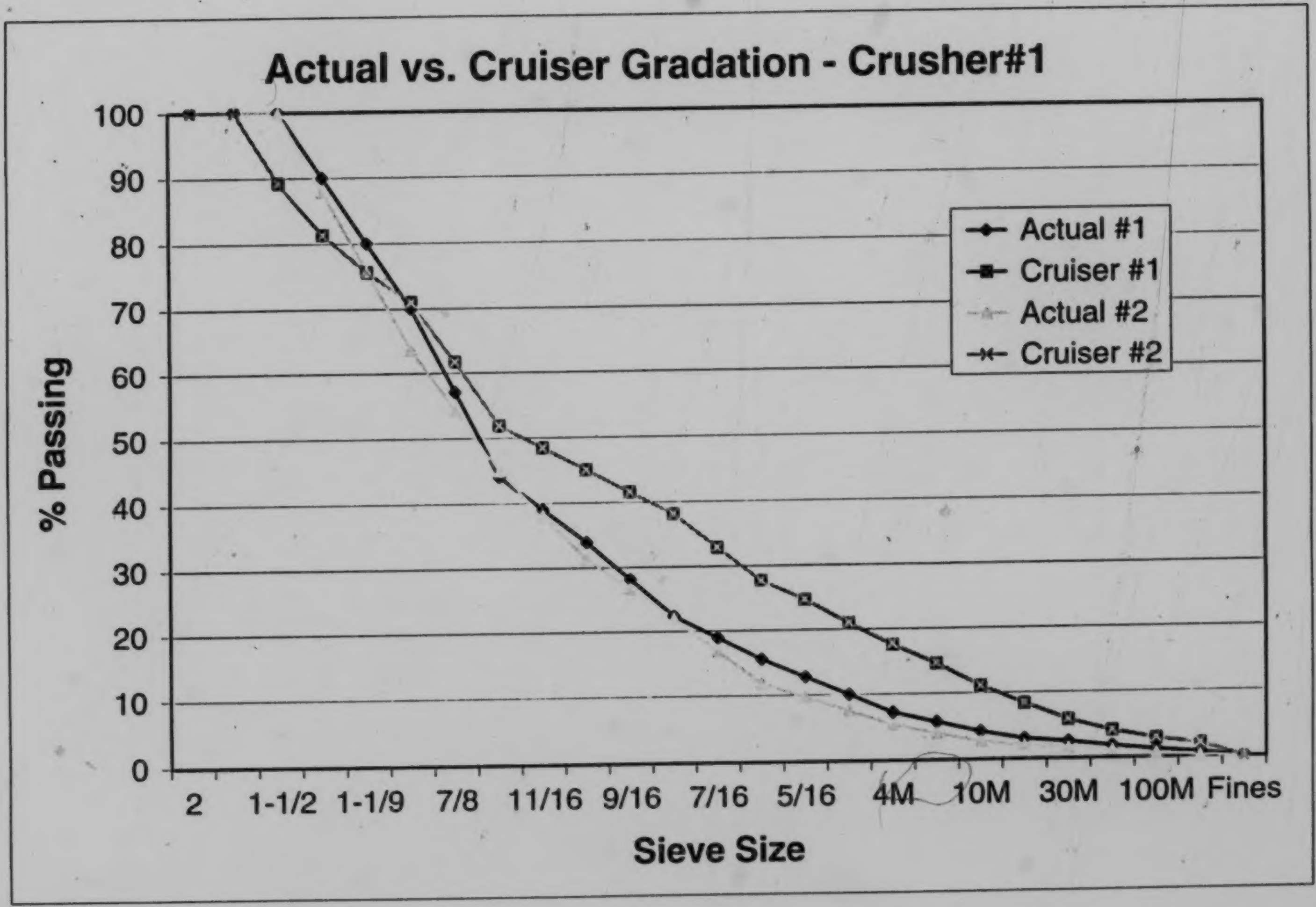


Actual Vs. Cruiser Results For All Four Samples (Crusher #1)

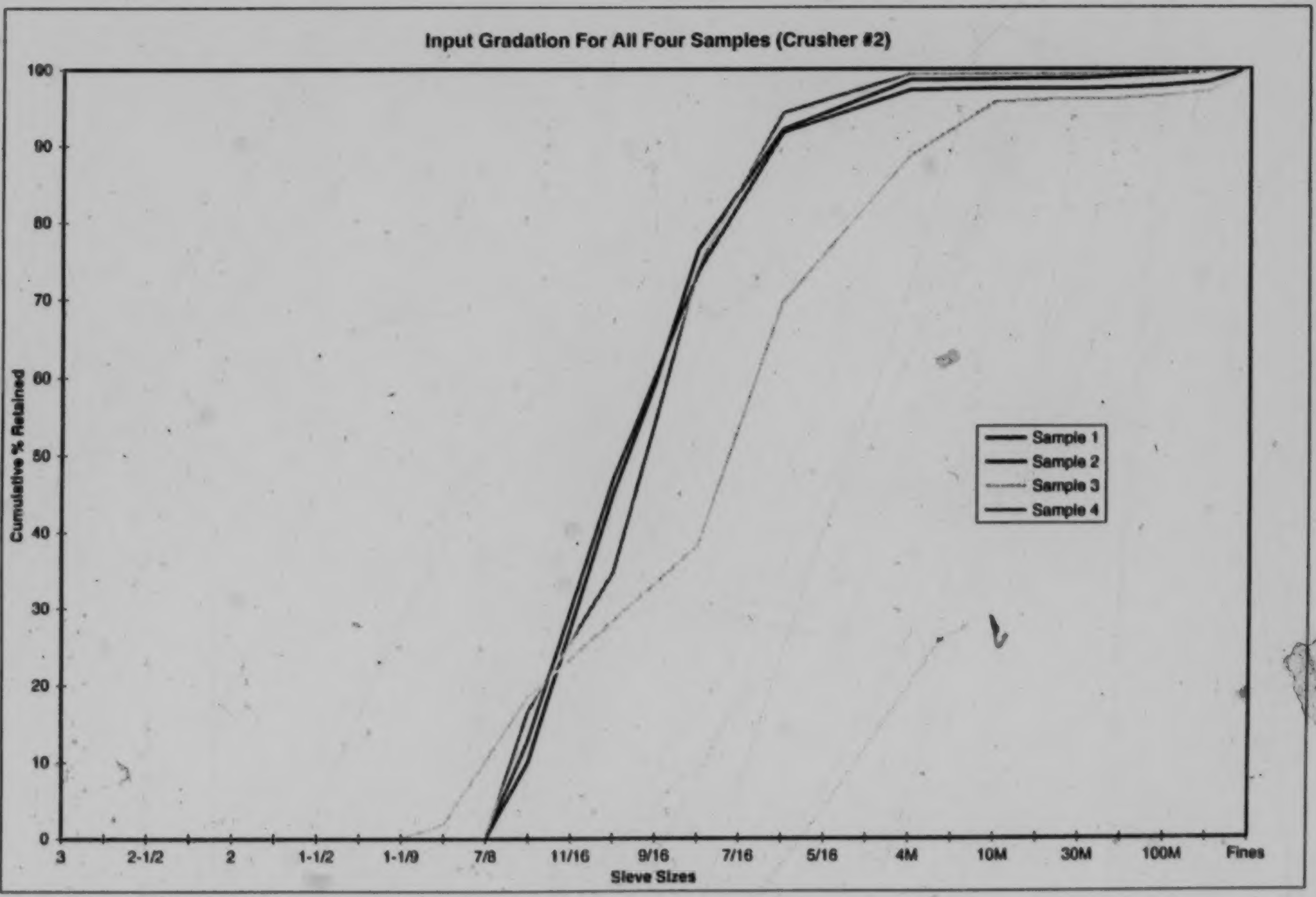


Actual Vs. Cruiser Results For 2 Samples (Crusher #1)

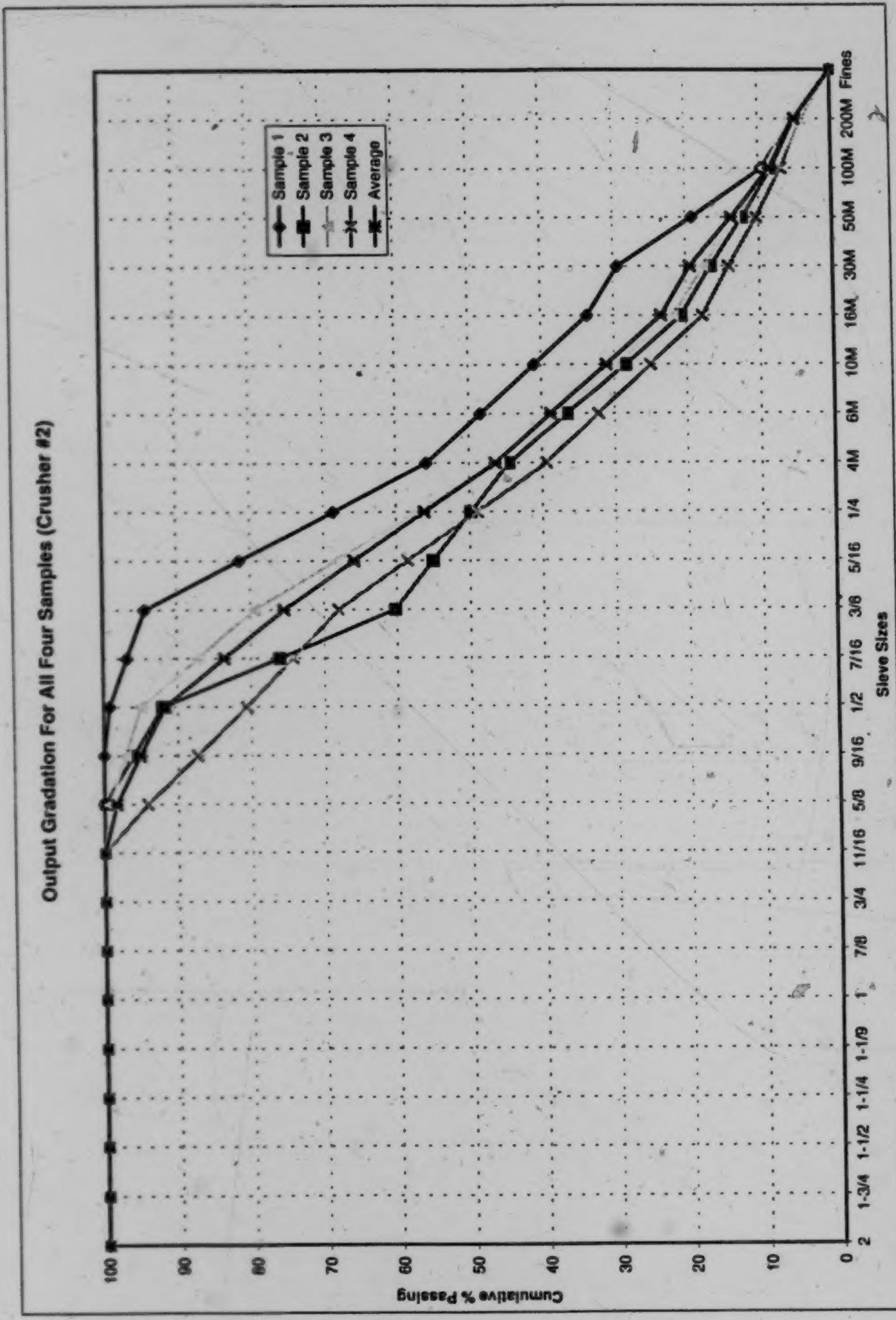

Actual vs. Cruiser Gradation - Crusher#1
% Passing
100
90
80
70
60
50
40
30
20
10
0
Sieve Size
2
1-1/2
1-1/9
7/8
11/16
9/16
7/16
5/16
4M
10M
30M
100M
Fines
Actual #1
Cruiser #1
Actual #2
Cruiser #2
Actual #3
Cruiser #3
Actual #4
Cruiser #4
Actual Avg.
Cruiser Avg.

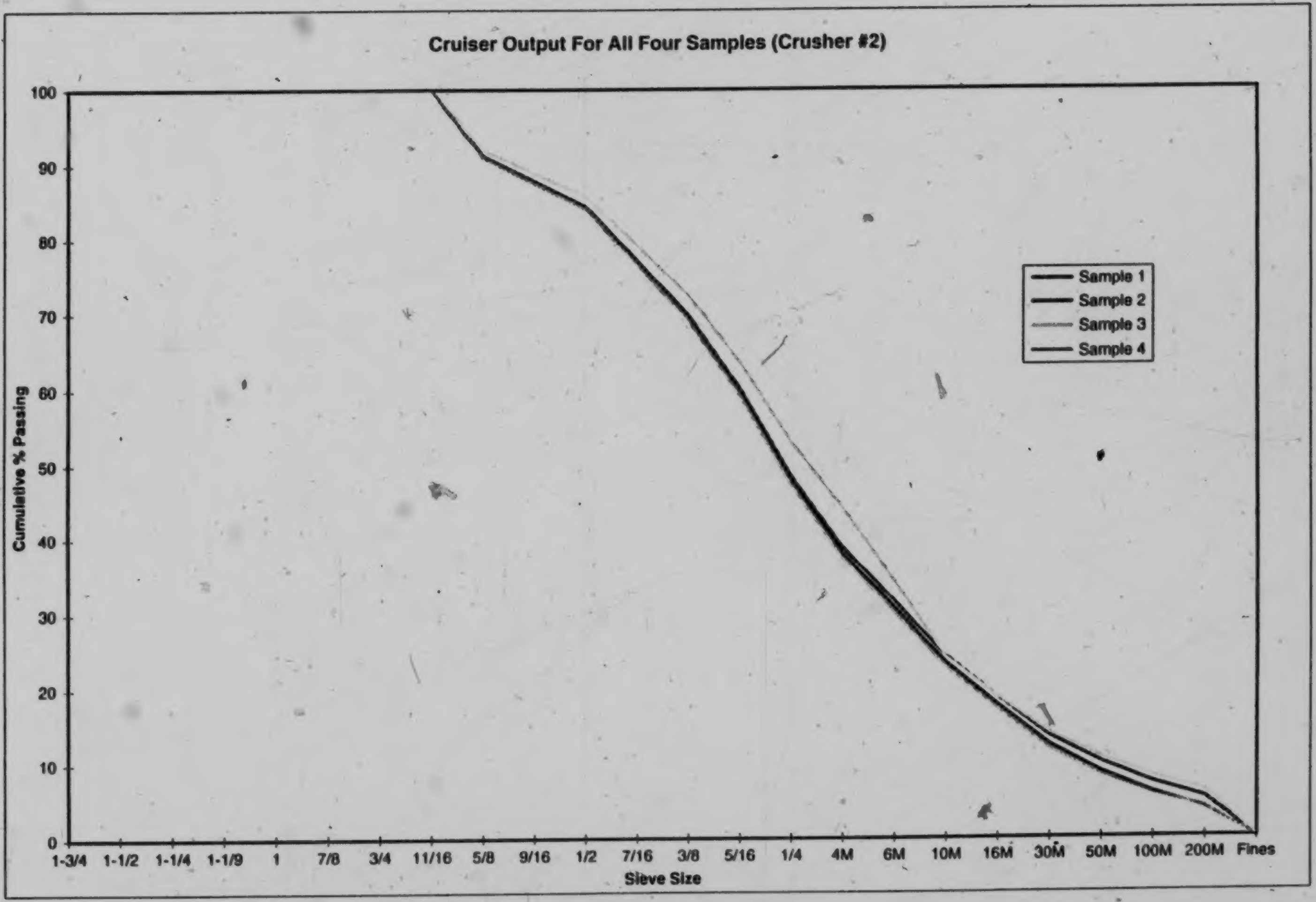

Actual vs. Cruiser Gradation - Crusher#1
% Passing
100
90
80
70
60
50
40
30
20
10
0
2
1-1/2
1-1/9
7/8
11/16
9/16
7/16
5/16
4M
10M
30M
100M
Fines
Sieve Size
Actual #1
Cruiser #1
Actual #2
Cruiser #2

APPENDIX I

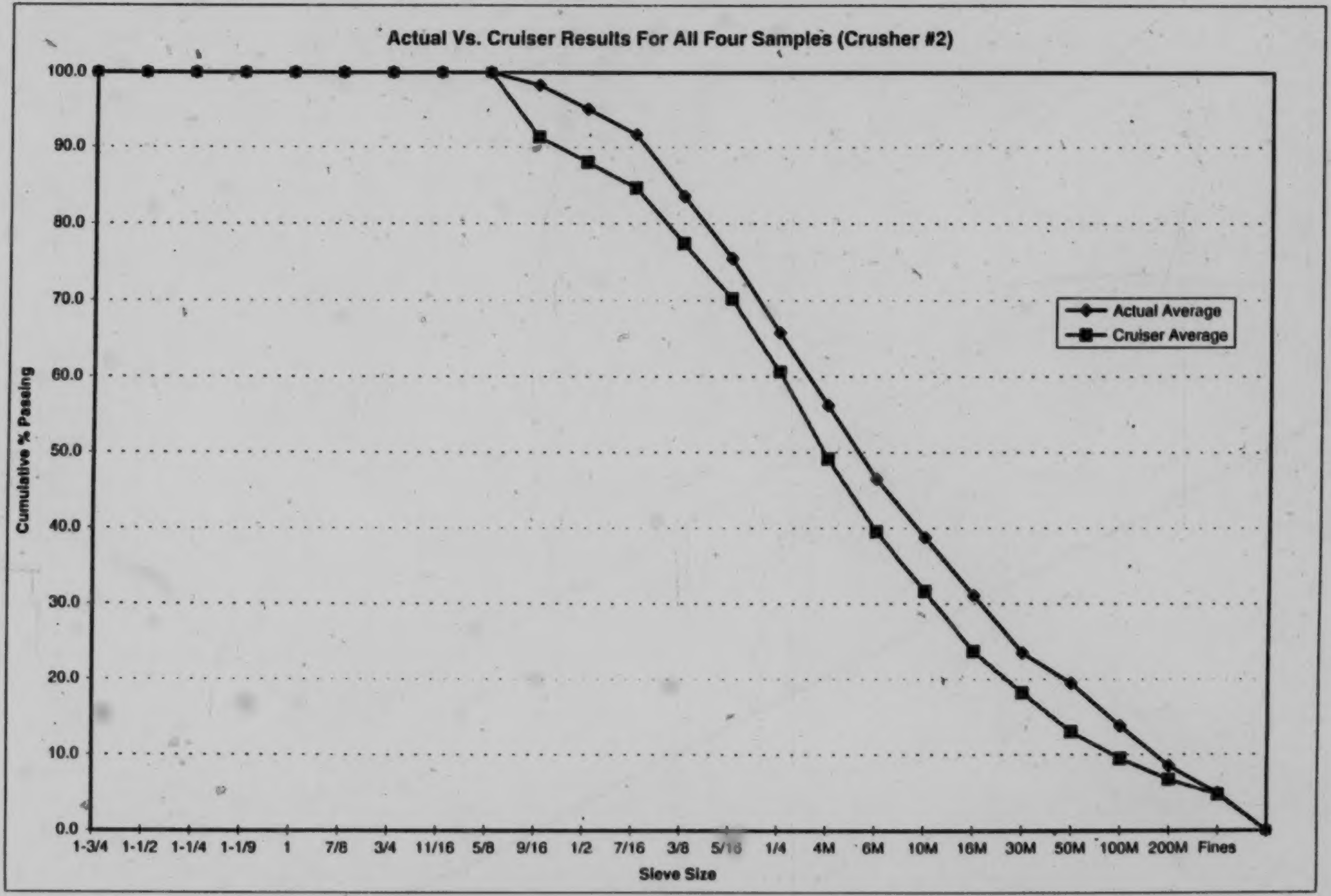


Input Gradation For All Four Samples (Crusher #2)

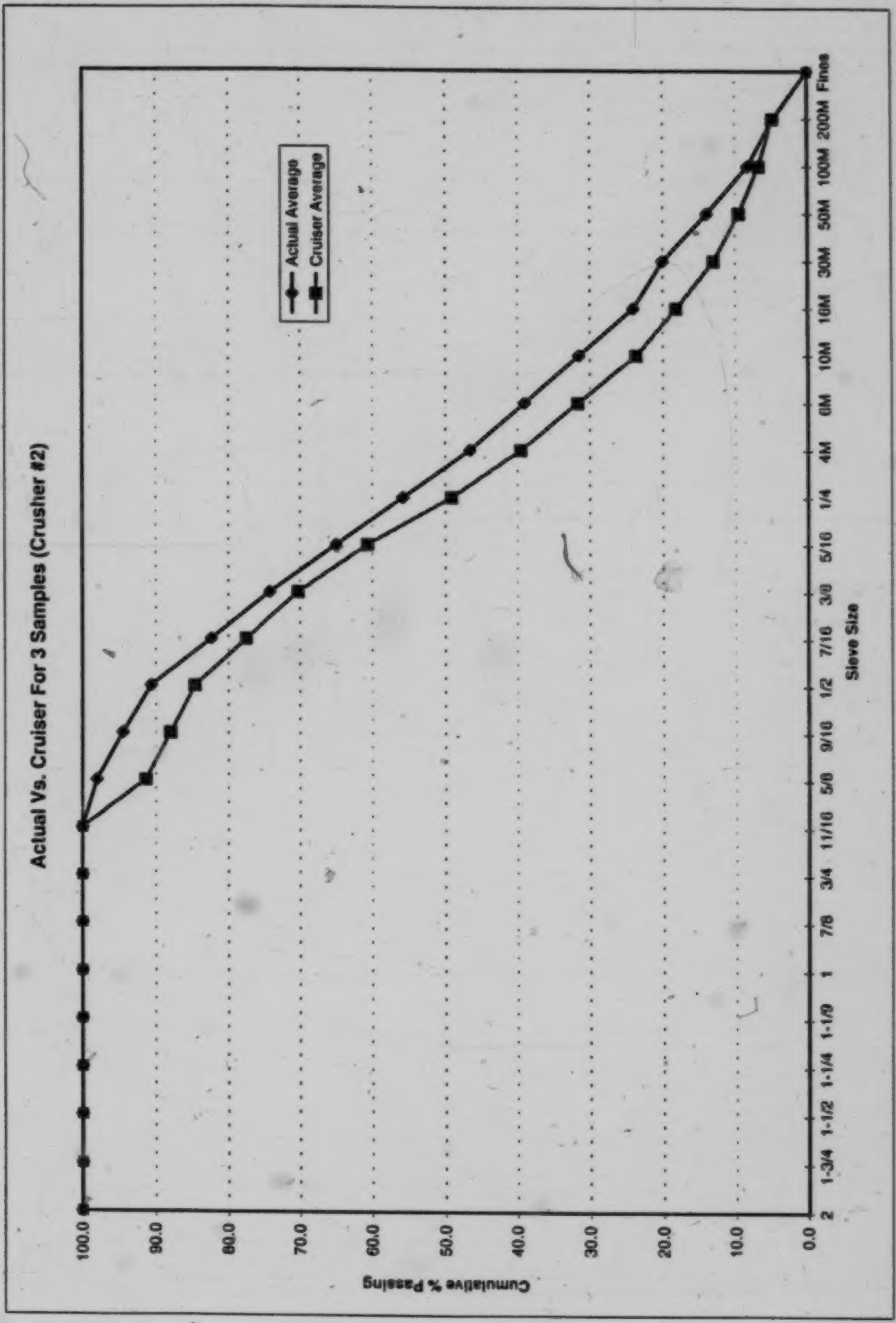

Output Gradation For All Four Samples (Crusher #2)
Sample 1
Sample 2
Sample 3
Sample 4
Average
Cumulative % Passing
100
90
80
70
60
50
40
30
20
10
0
Sieve Sizes
2
1-3/4
1-1/2
1-1/4
1-1/9
1
7/8
3/4
11/16
5/8
9/16
1/2
7/16
3/8
5/16
1/4
4M
6M
10M
16M
30M
50M
100M
200M
Fines

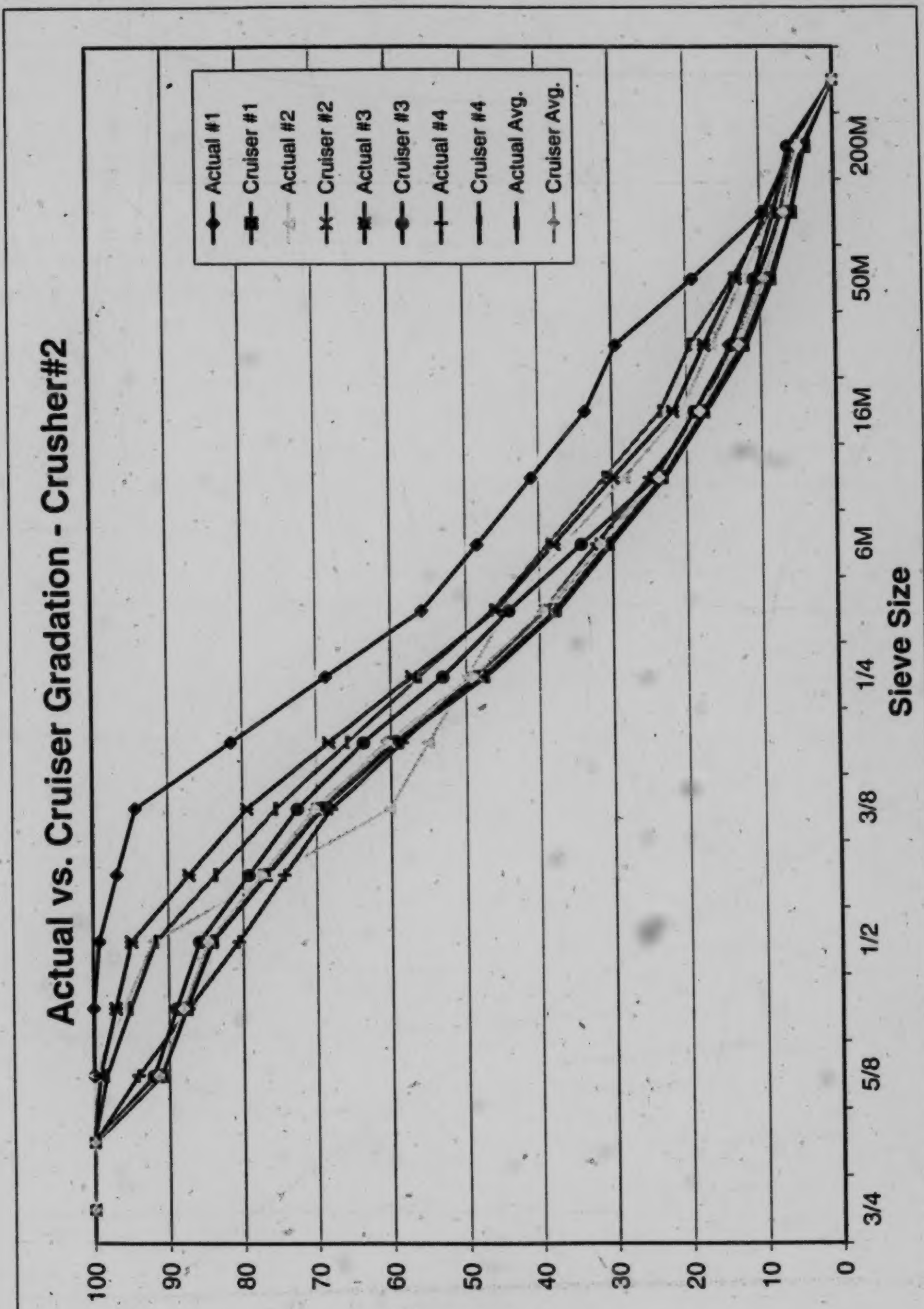


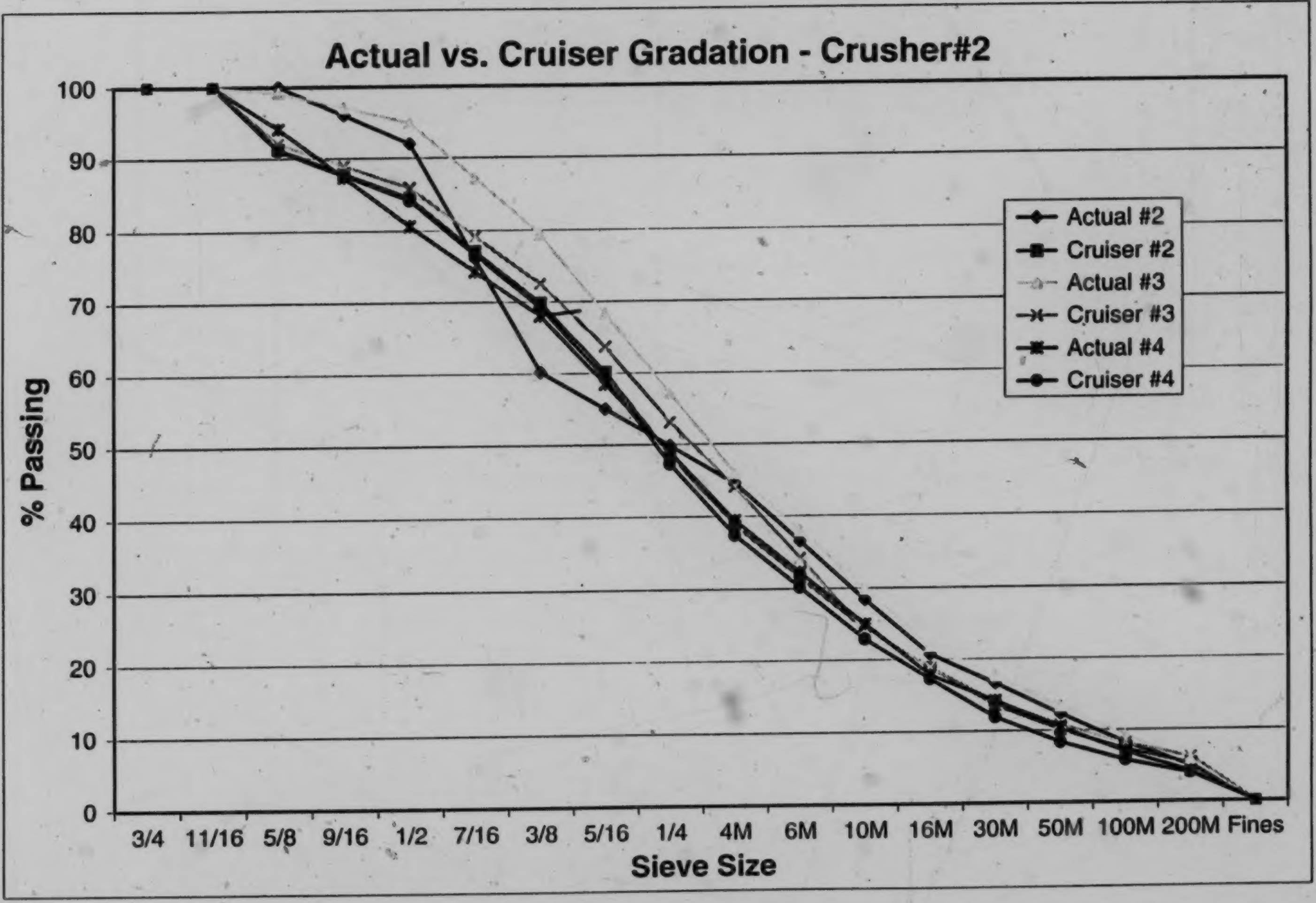
Cruiser Output For All Four Samples (Crusher #2)



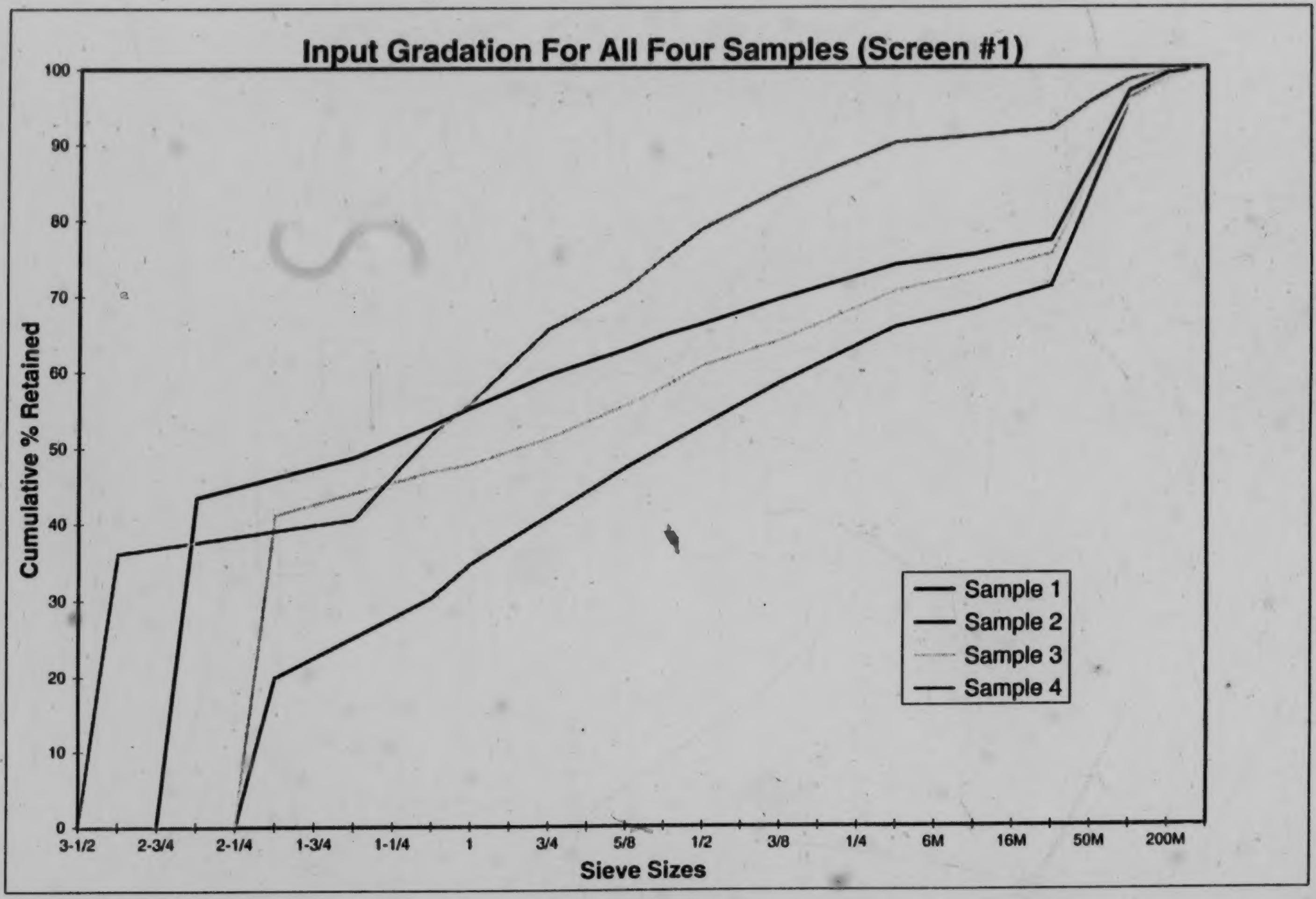
Actual Vs. Cruiser Results For All Four Samples (Crusher #2)


Actual Vs. Cruiser For 3 Samples (Crusher #2)
Actual Average
Cruiser Average
Cumulative % Passing
100.0
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
2
1-3/4
1-1/2
1-1/4
1-1/8
1
7/8
3/4
11/16
5/8
9/16
1/2
7/16
3/8
5/16
1/4
4M
8M
10M
16M
30M
50M
100M
200M
Fines
Sieve Size

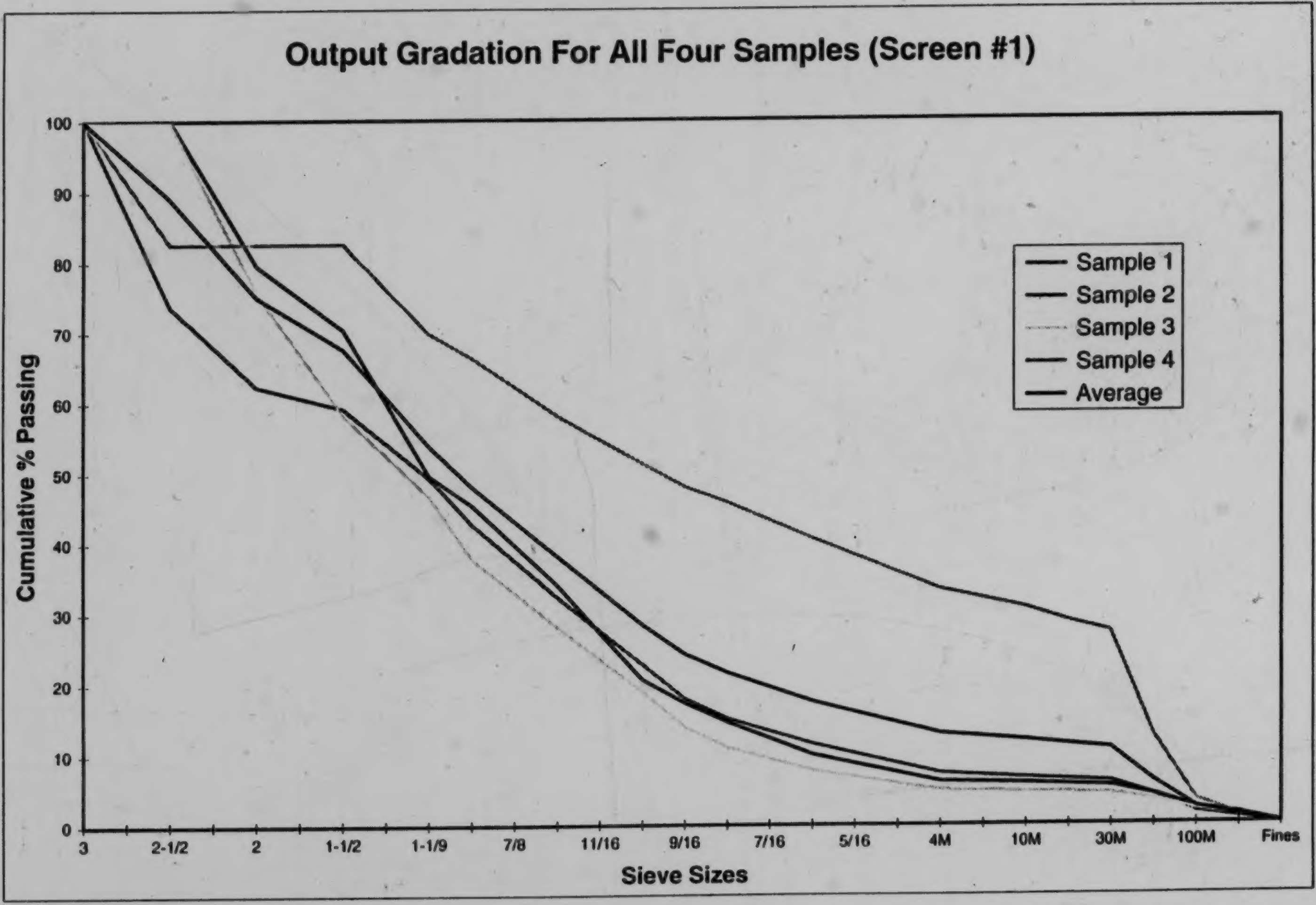

Actual vs. Cruiser Gradation - Crusher#2
% Passing
100
90
80
70
60
50
40
30
20
10
0
Sieve Size
3/4
5/8
1/2
3/8
1/4
6M
16M
50M
200M
Actual #1
Cruiser #1
Actual #2
Cruiser #2
Actual #3
Cruiser #3
Actual #4
Cruiser #4
Actual Avg.
Cruiser Avg.

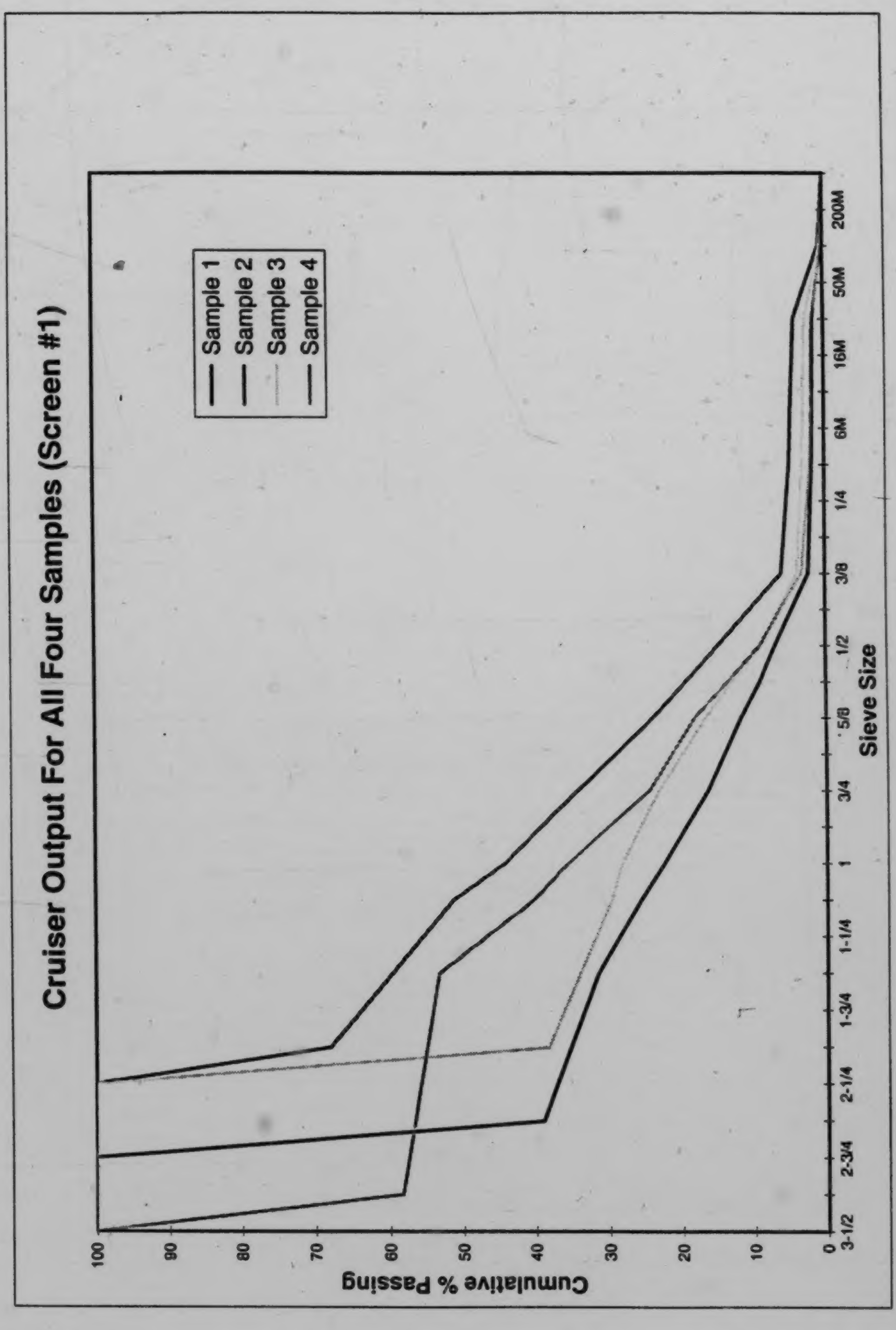

Actual vs. Cruiser Gradation - Crusher#2

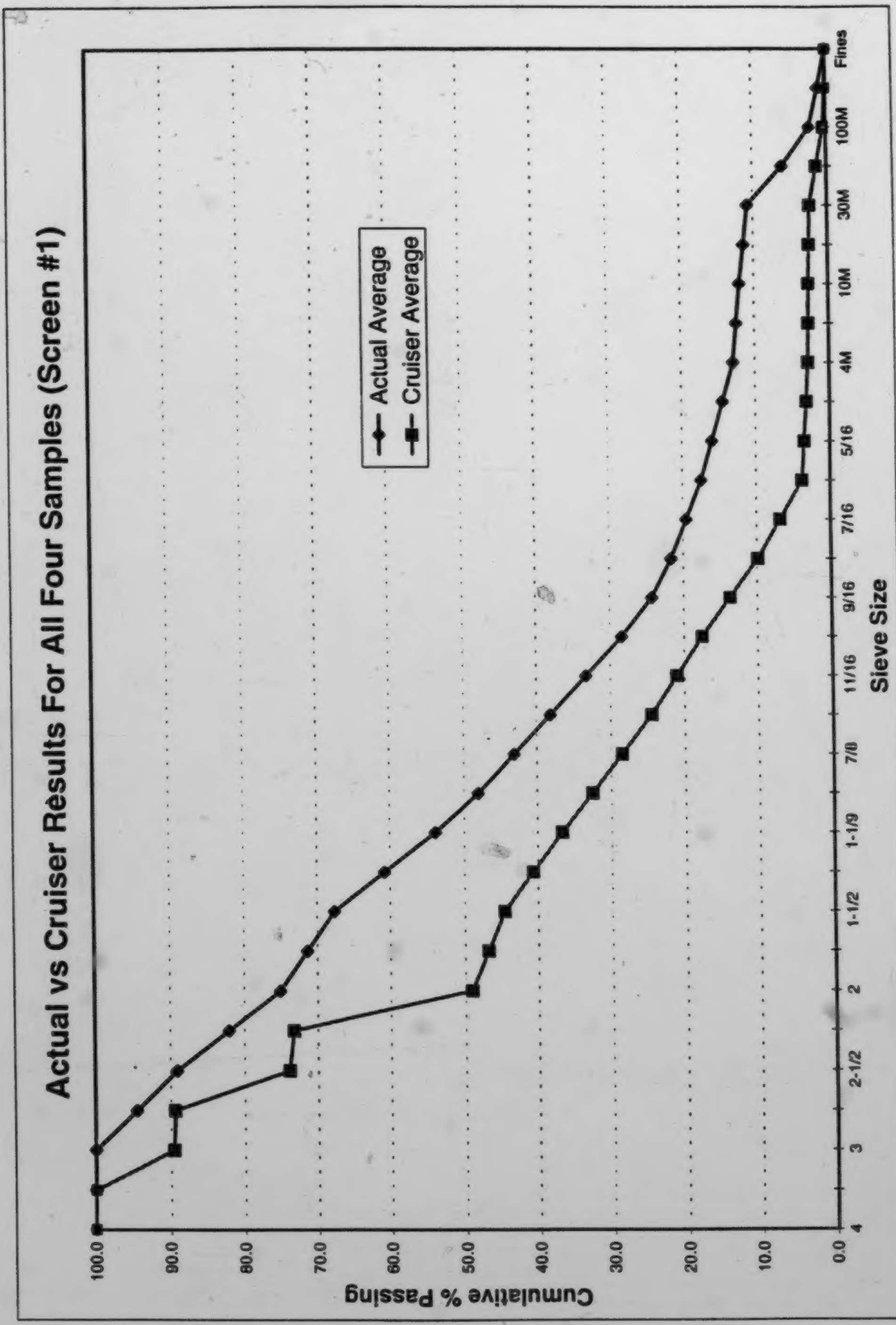
APPENDIX J

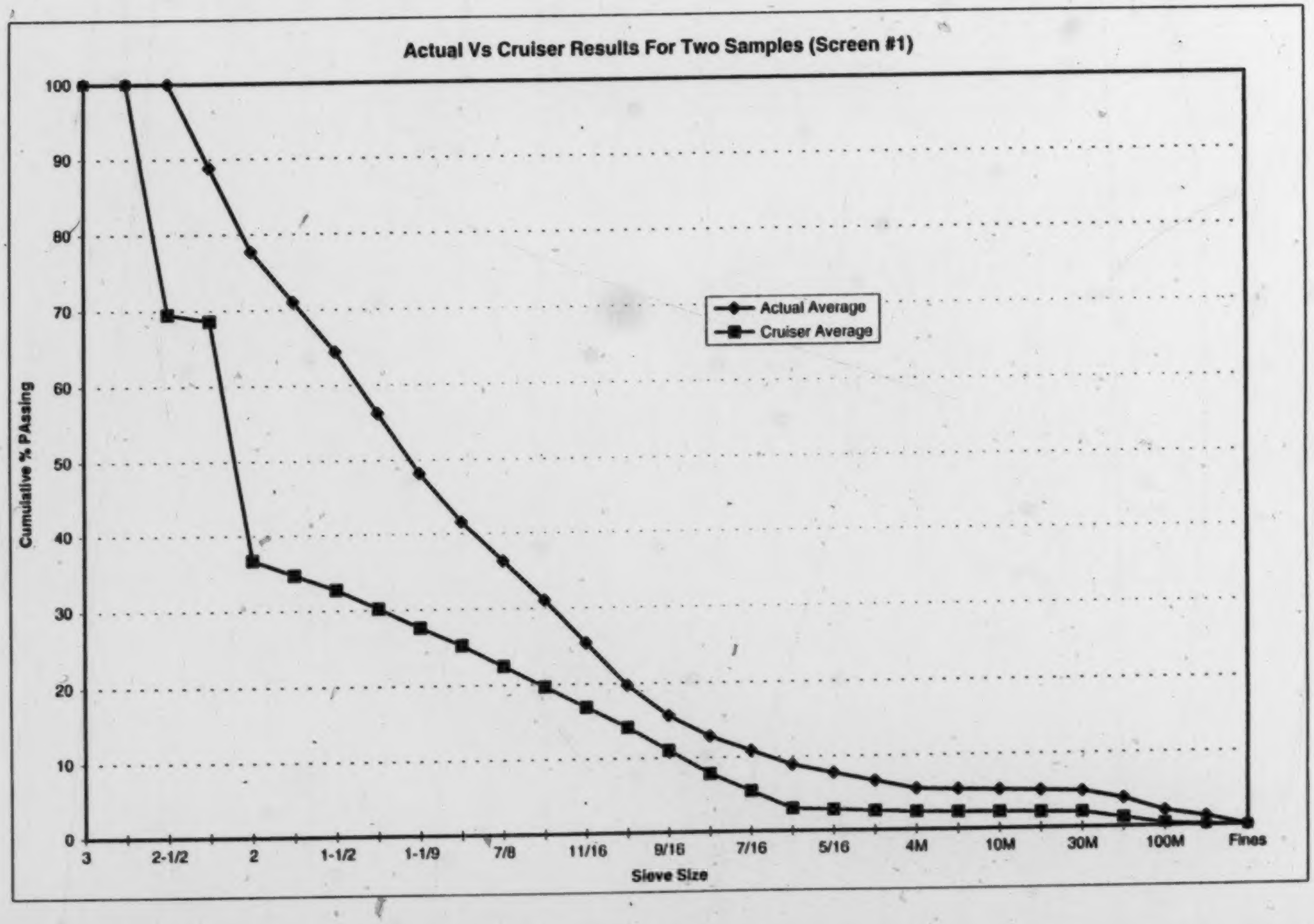


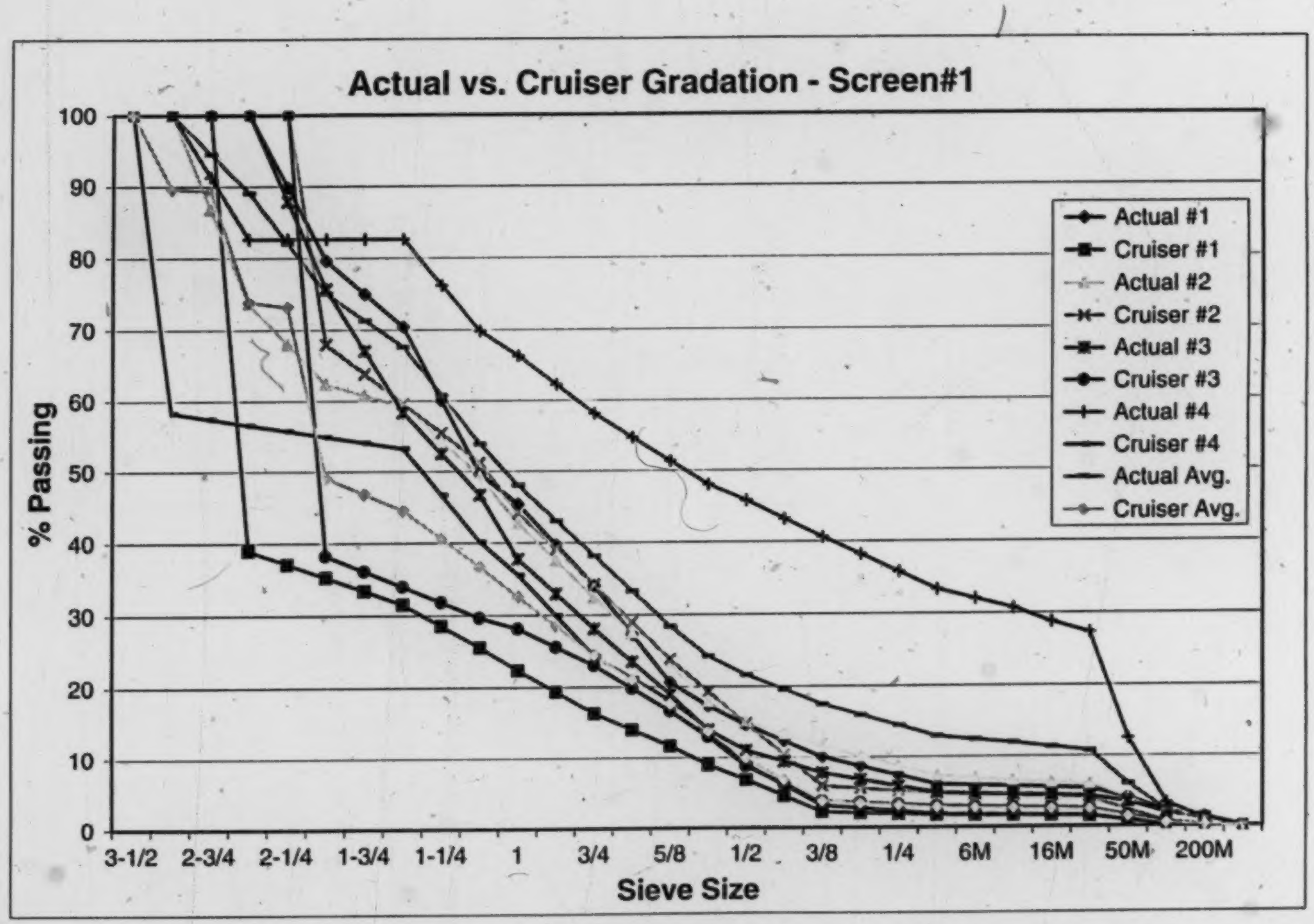
Input Gradation For All Four Samples (Screen #1)

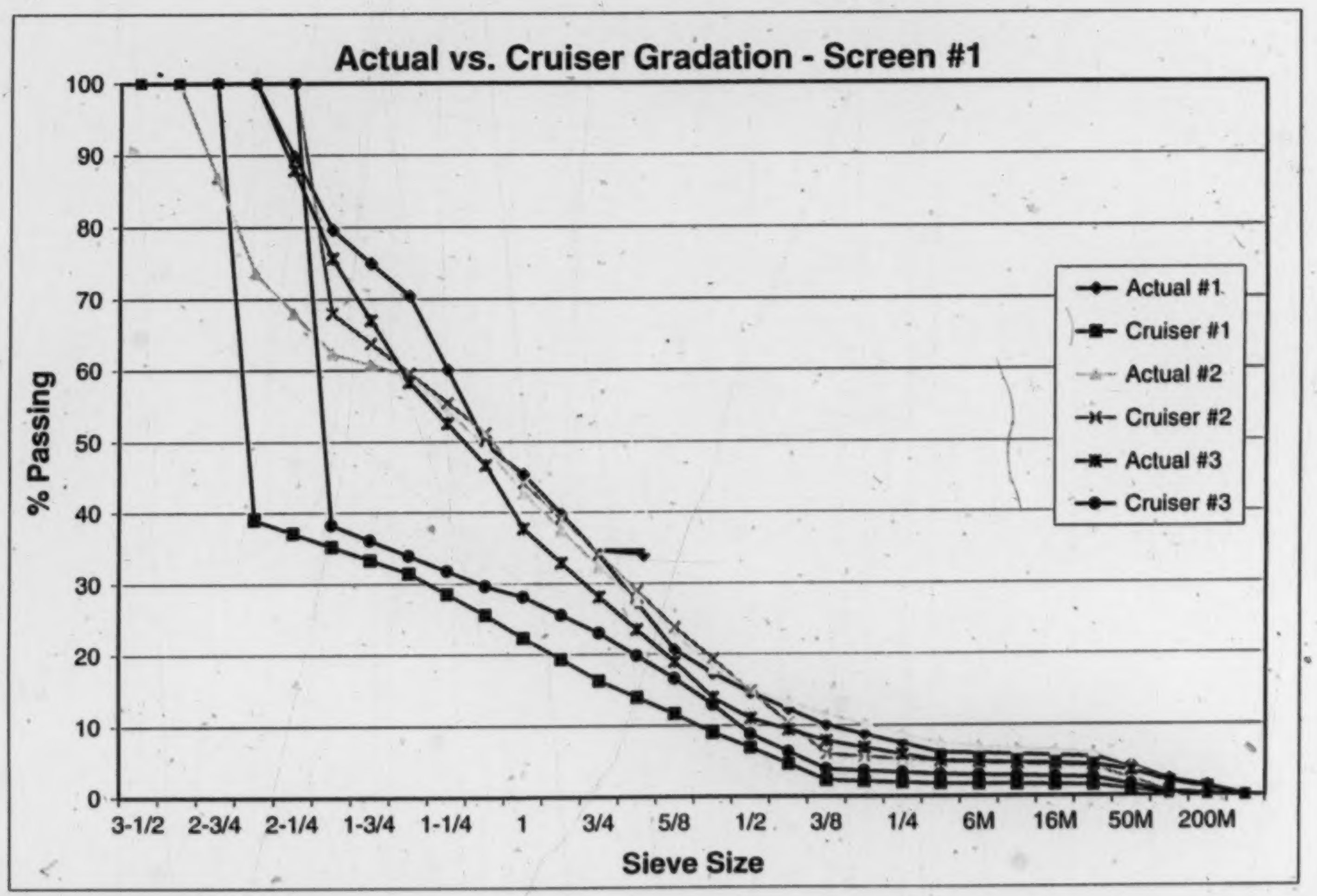

Output Gradation For All Four Samples (Screen #1)


Cruiser Output For All Four Samples (Screen #1)
Sample 1
Sample 2
Sample 3
Sample 4
Cumulative % Passing
100
90
80
70
60
50
40
30
20
10
0
3-1/2
2-3/4
2-1/4
1-3/4
1-1/4
1
3/4
5/8
1/2
3/8
1/4
6M
16M
50M
200M
Sieve Size


Actual vs Cruiser Results For All Four Samples (Screen #1)
Actual Average
Cruiser Average
Cumulative % Passing
100.0
90.0
80.0
70.0
60.0
50.0
40.0
30.0
20.0
10.0
0.0
4
3
2-1/2
2
1-1/2
1-1/8
7/8
11/16
9/16
7/16
5/16
4M
10M
30M
100M
Fines
Sieve Size

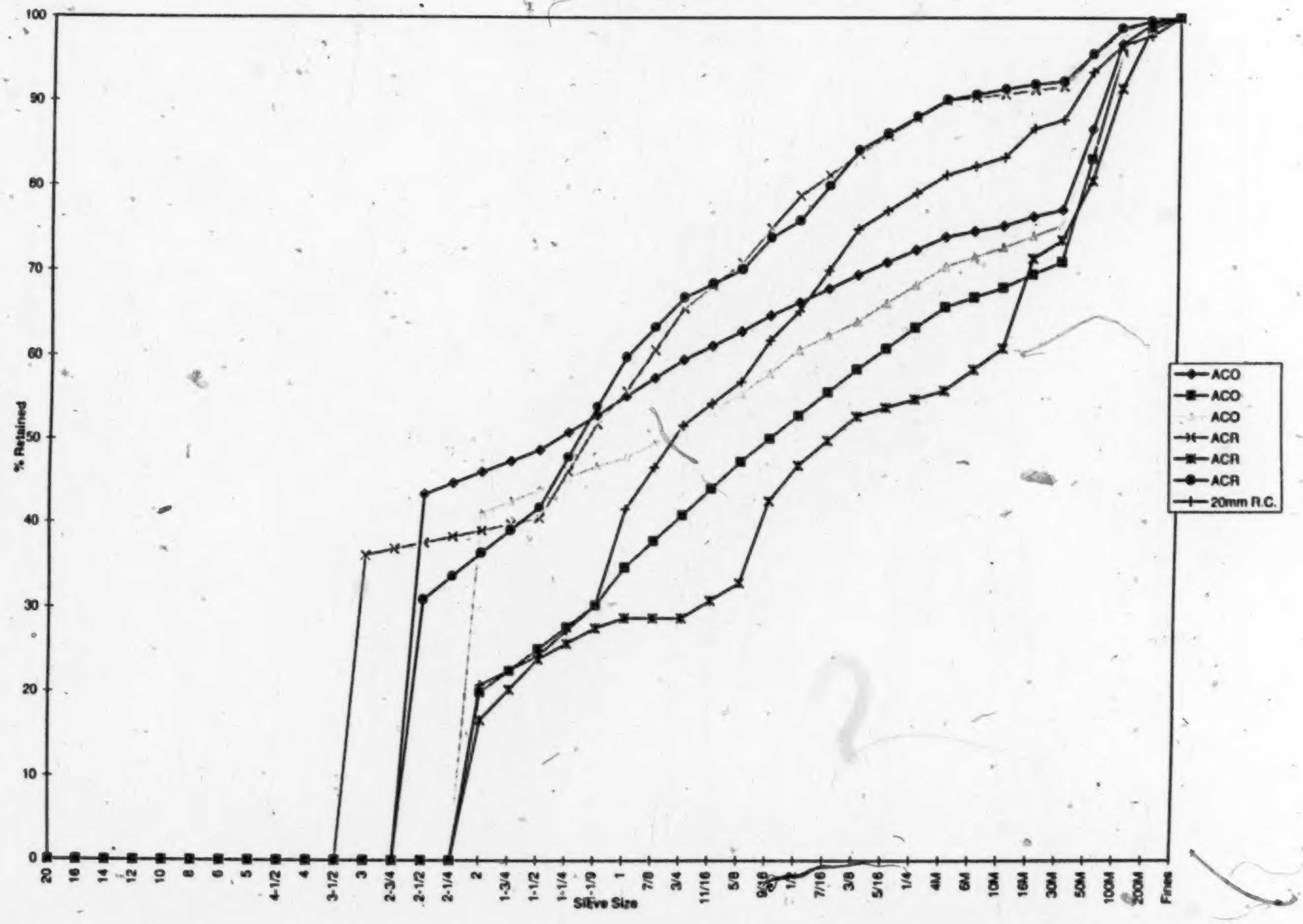

Actual Vs Cruiser Results For Two Samples (Screen #1)
Cumulative % PAssing
100
90
80
70
60
50
40
30
20
10
0
3
2-1/2
2
1-1/2
1-1/9
7/8
11/16
9/16
7/16
5/16
4M
10M
30M
100M
Fines
Sieve Size
Actual Average
Cruiser Average


Actual vs. Cruiser Gradation - Screen#1

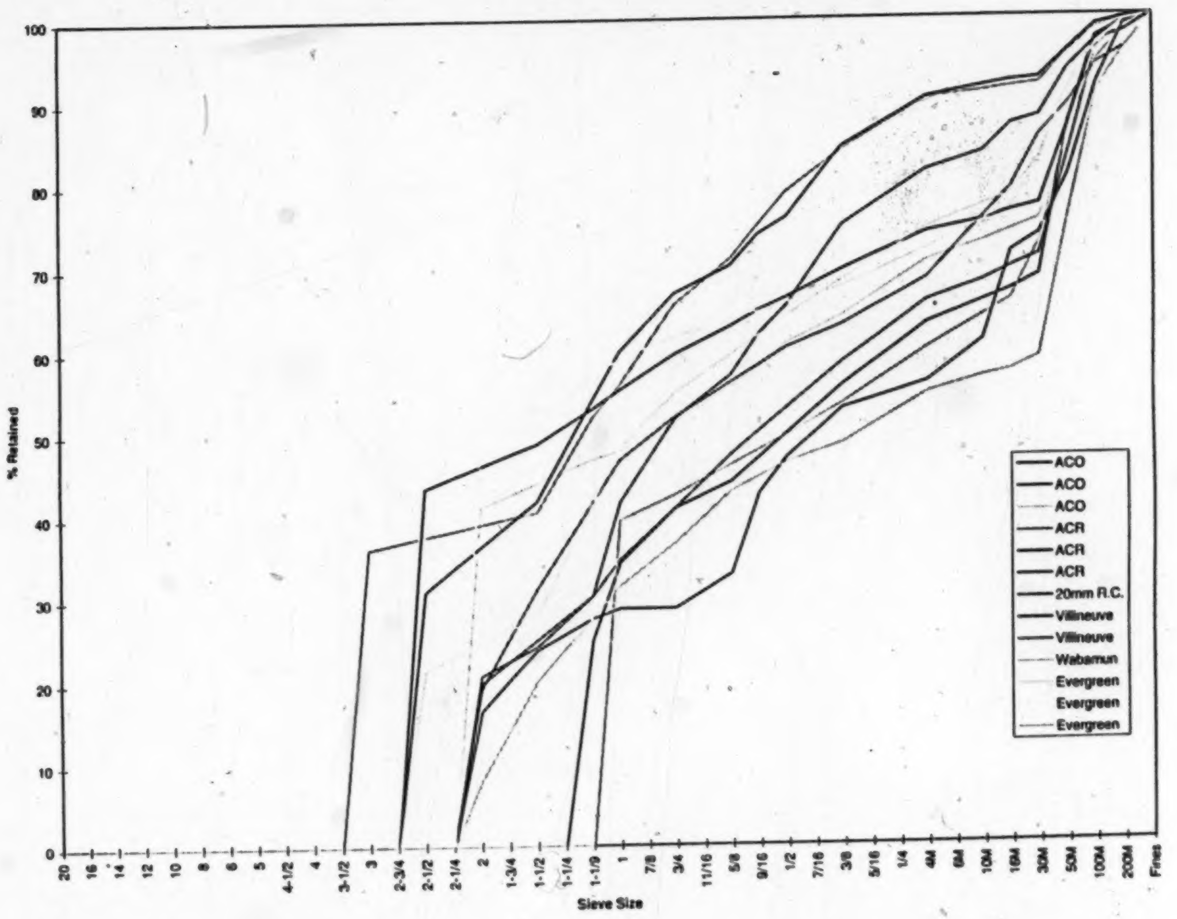

Actual vs. Cruiser Gradation - Screen #1
% Passing
100
90
80
70
60
50
40
30
20
10
0
3-1/2
2-3/4
2-1/4
1-3/4
1-1/4
1
3/4
5/8
1/2
3/8
1/4
6M
16M
50M
200M
Sieve Size
Actual #1
Cruiser #1
Actual #2
Cruiser #2
Actual #3
Cruiser #3

APPENDIX K

Raw Input Gradation

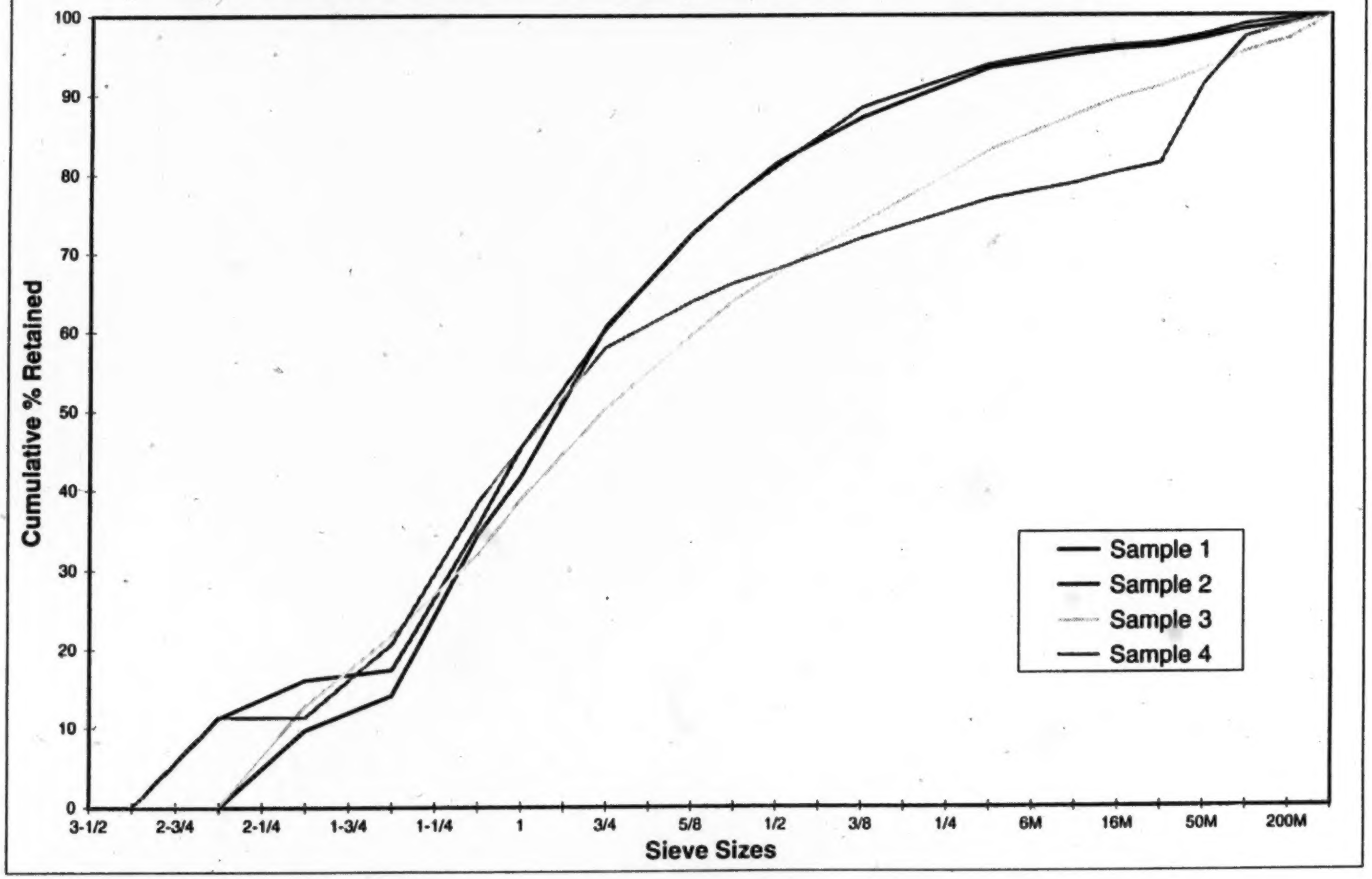


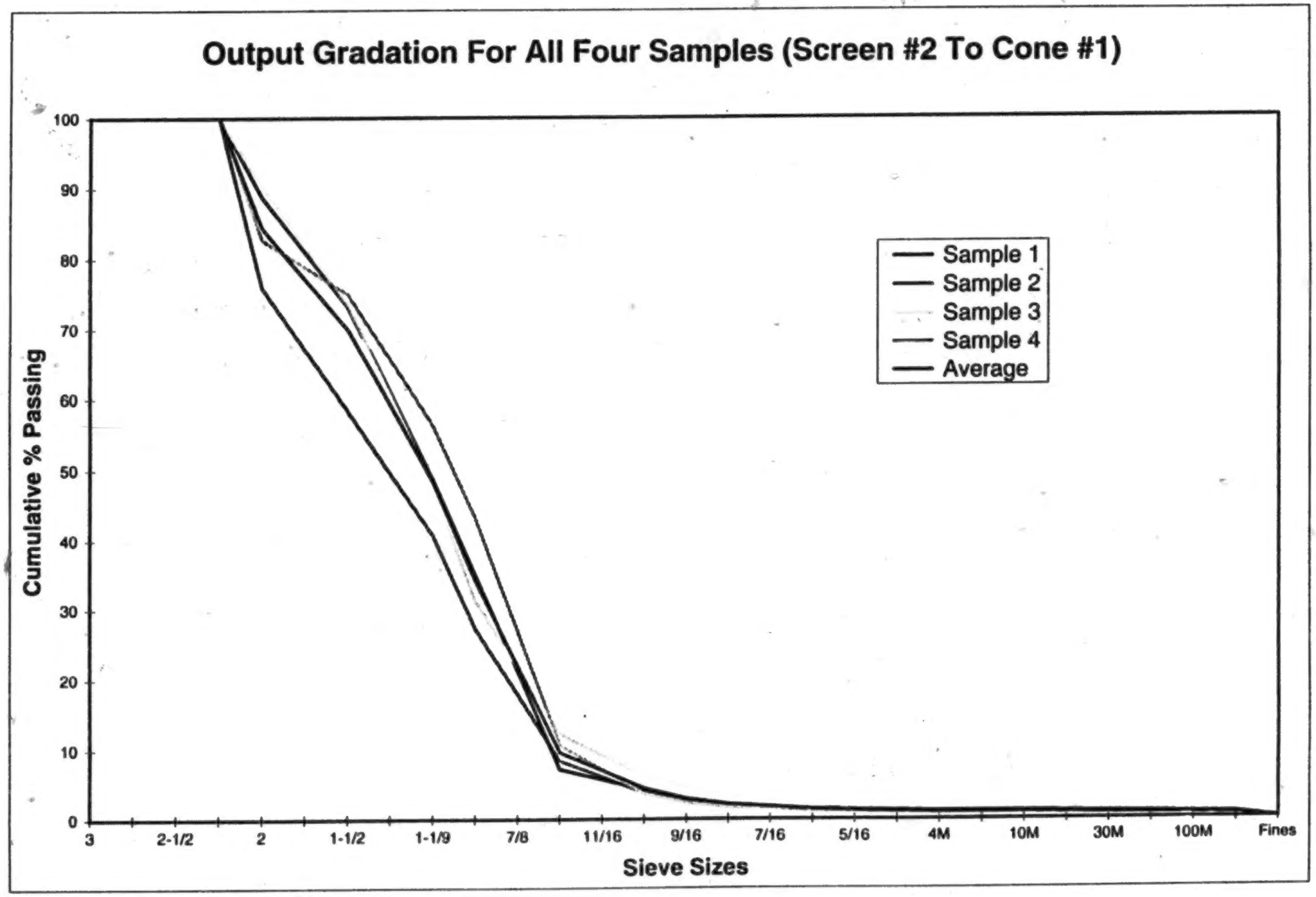
Input Gradation From Villeneuve and Other Plants



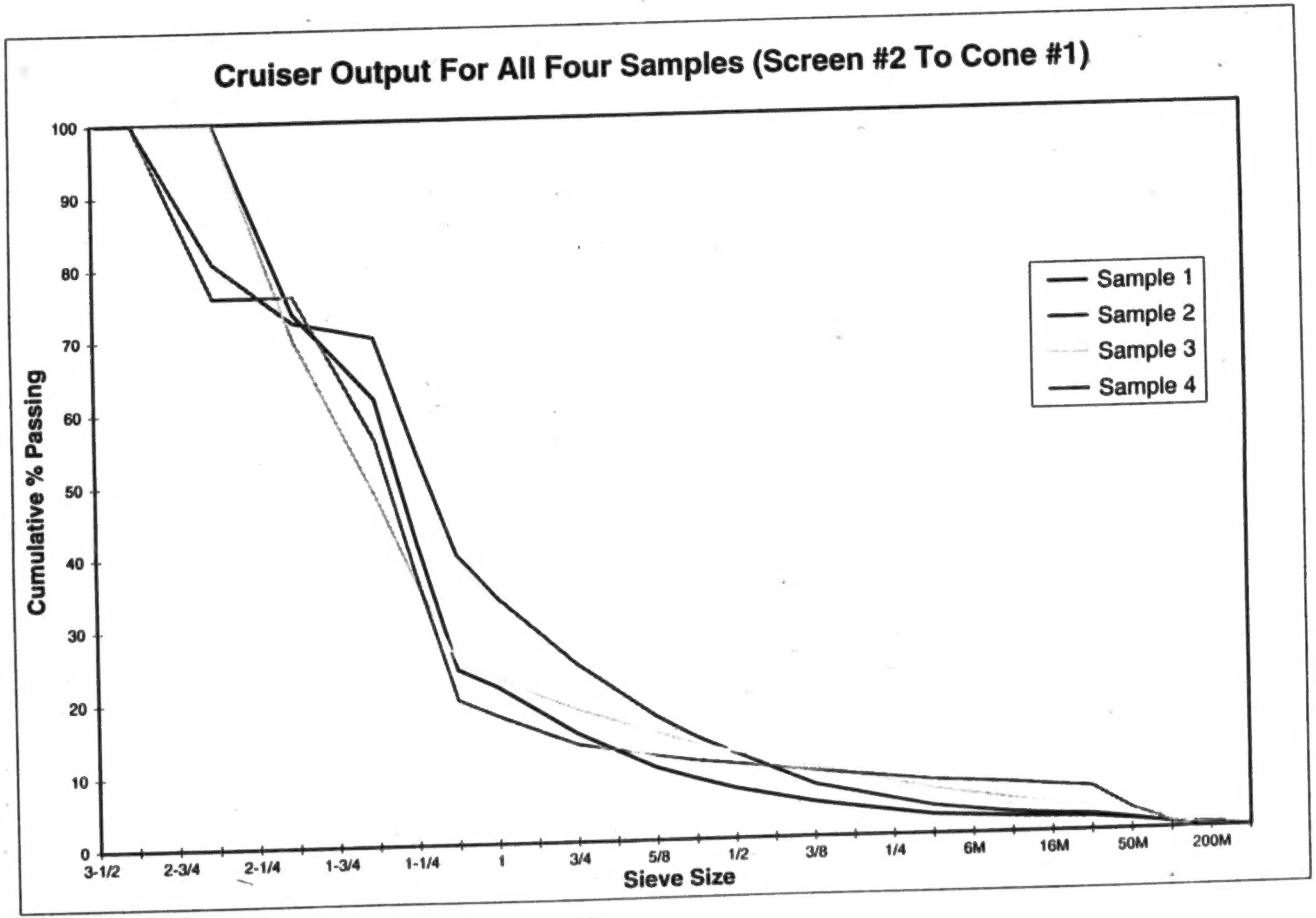
APPENDIX L

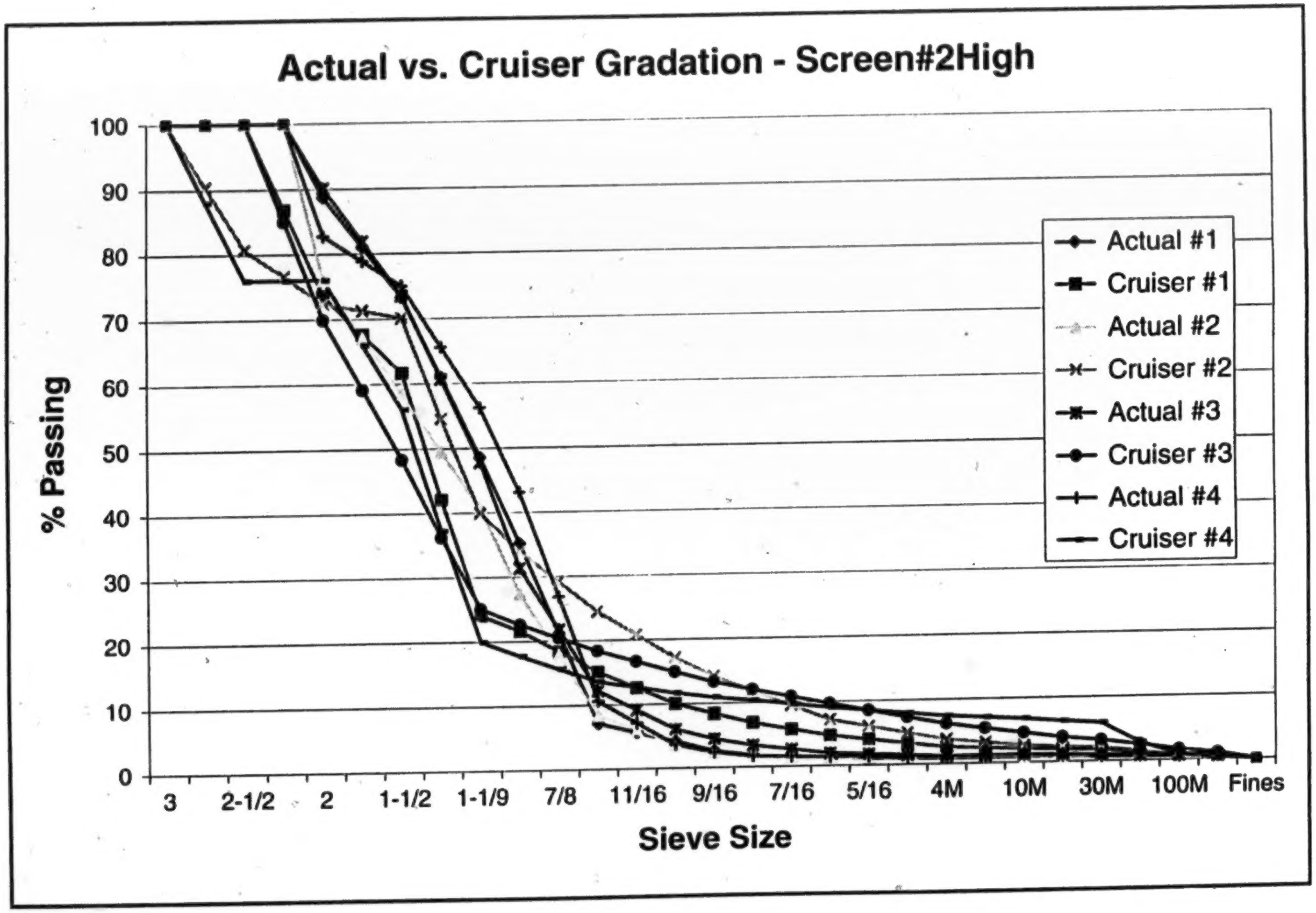
Input Gradation For All Four Samples (Screen #2 To Cone #1)



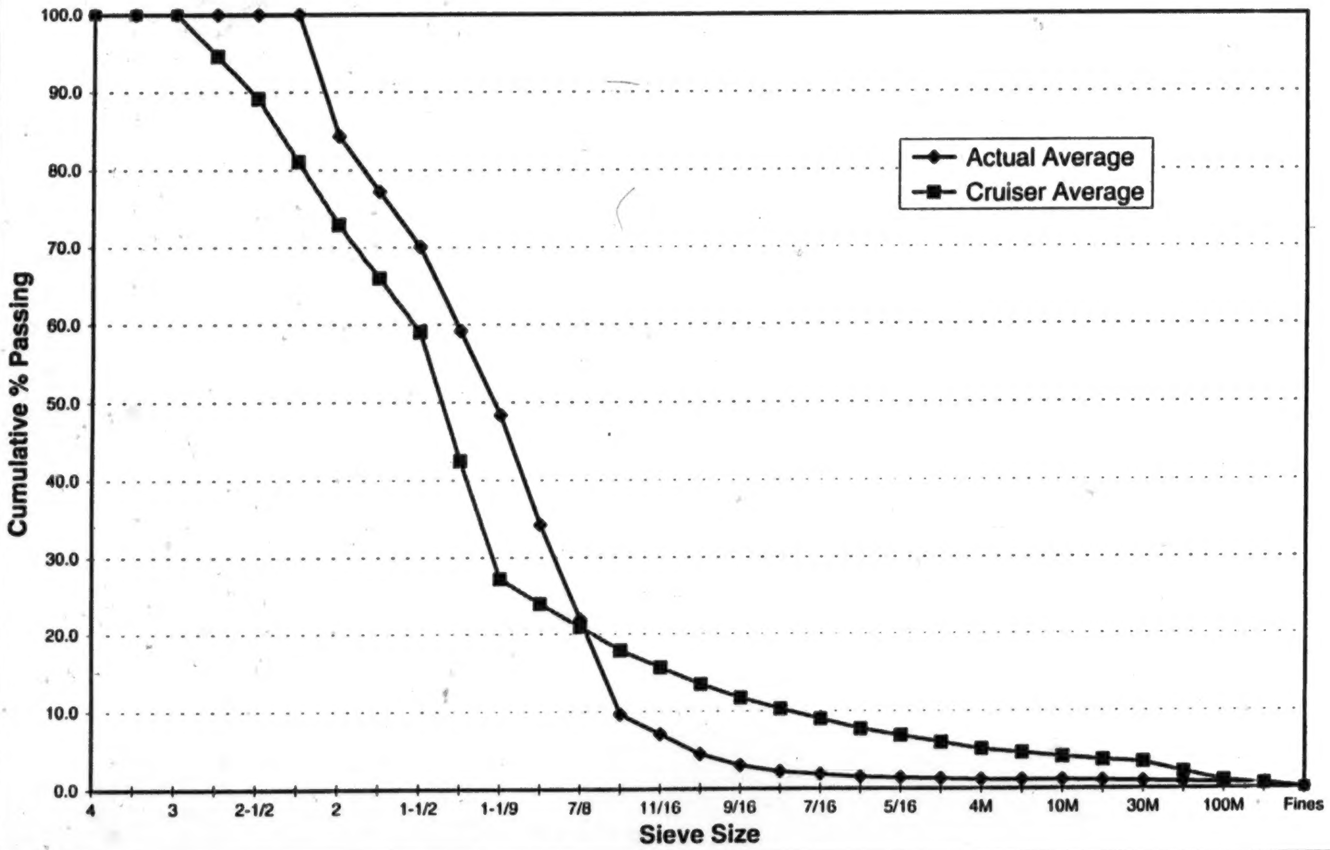


Output Gradation For All Four Samples (Screen #2 To Cone #1)

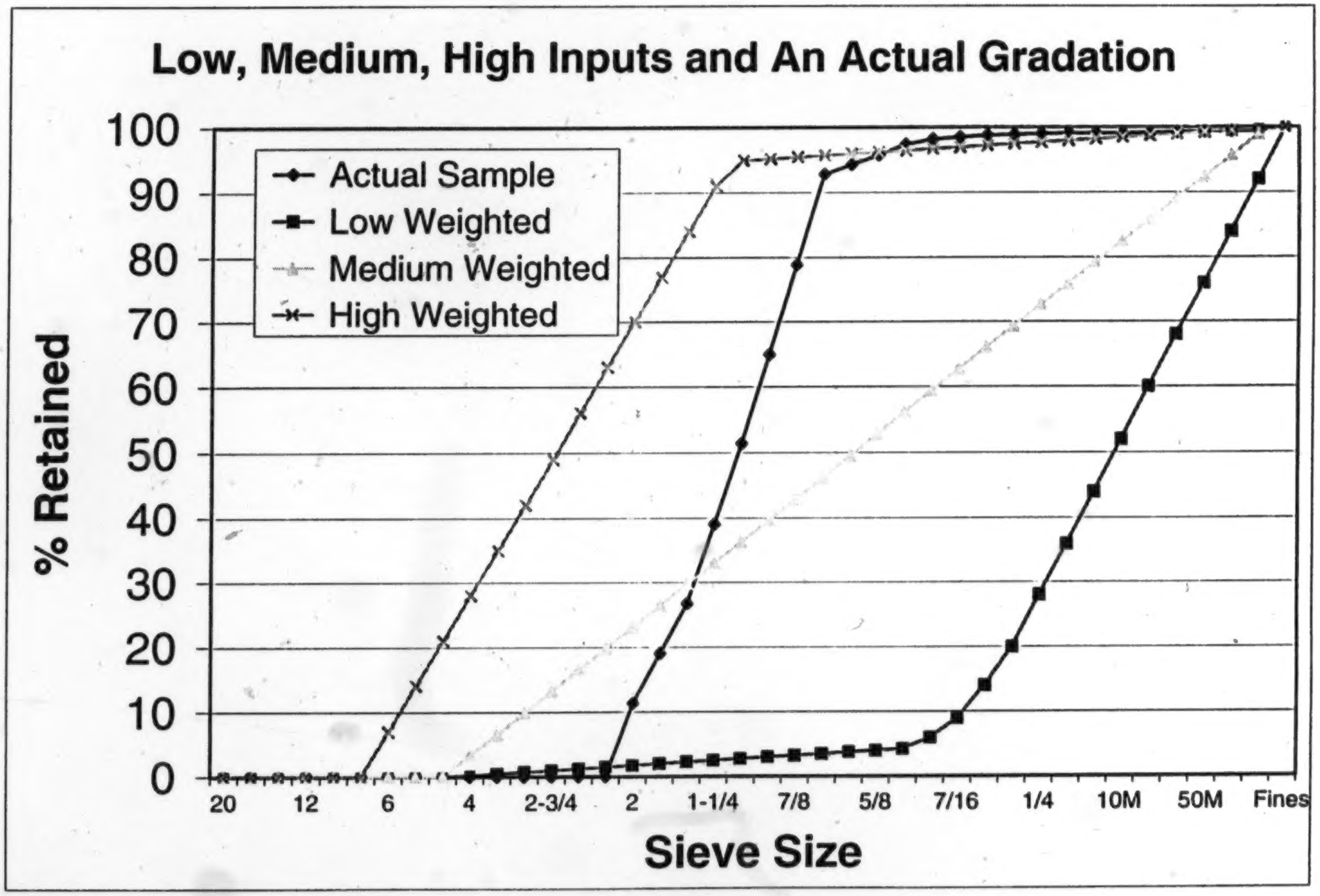

Cruiser Output For All Four Samples (Screen #2 To Cone #1)
Cumulative % Passing
100
90
80
70
60
50
40
30
20
10
0
3-1/2
2-3/4
2-1/4
1-3/4
1-1/4
1
3/4
5/8
1/2
3/8
1/4
6M
16M
50M
200M
Sieve Size
Sample 1
Sample 2
Sample 3
Sample 4


Actual vs. Cruiser Gradation - Screen#2High
% Passing
100
90
80
70
60
50
40
30
20
10
0
3
2-1/2
2
1-1/2
1-1/9
7/8
11/16
9/16
7/16
5/16
4M
10M
30M
100M
Fines
Sieve Size
Actual #1
Cruiser #1
Actual #2
Cruiser #2
Actual #3
Cruiser #3
Actual #4
Cruiser #4

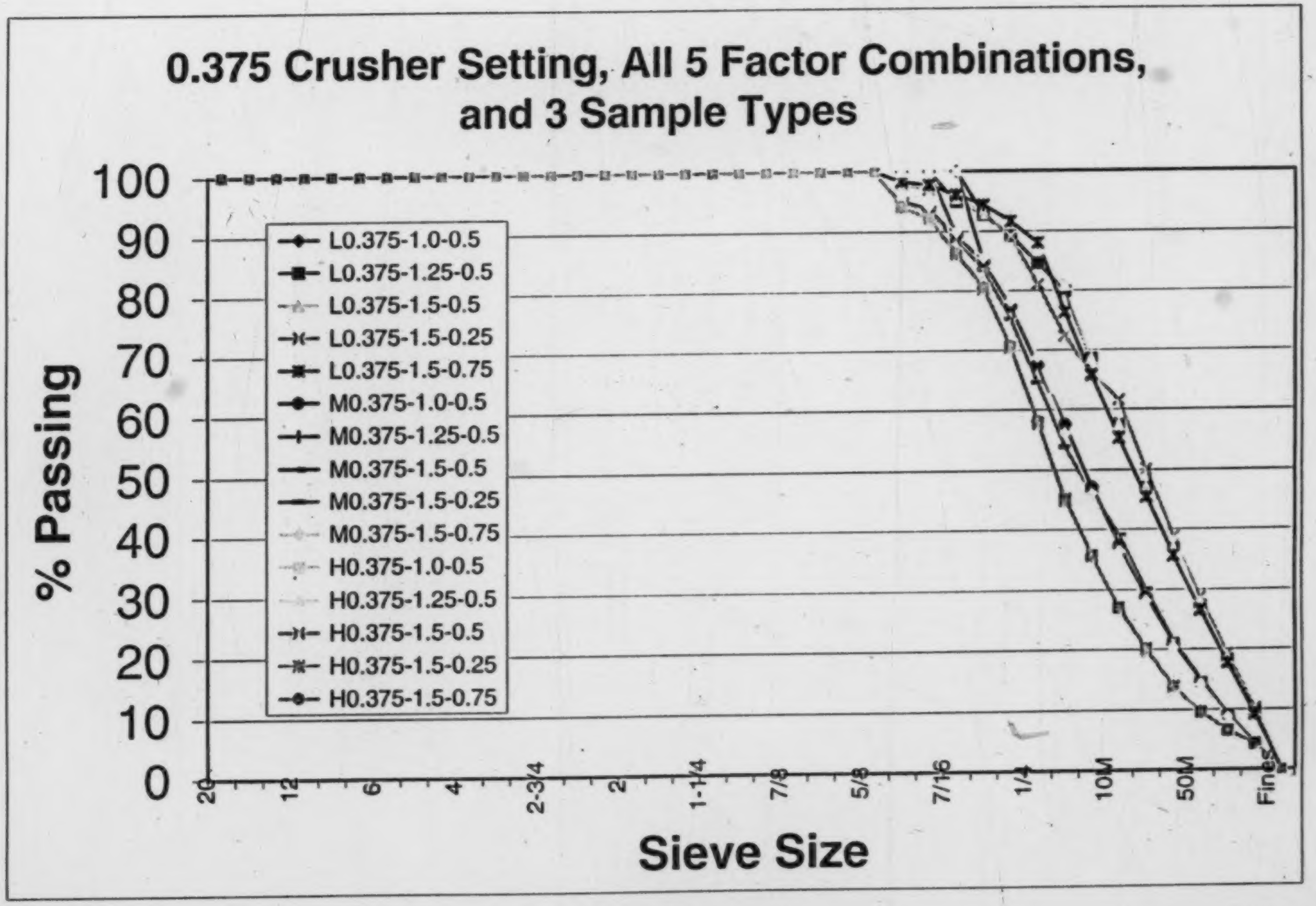
Actual vs Cruiser Results For Four Samples (Screen #2 To Cone #1)



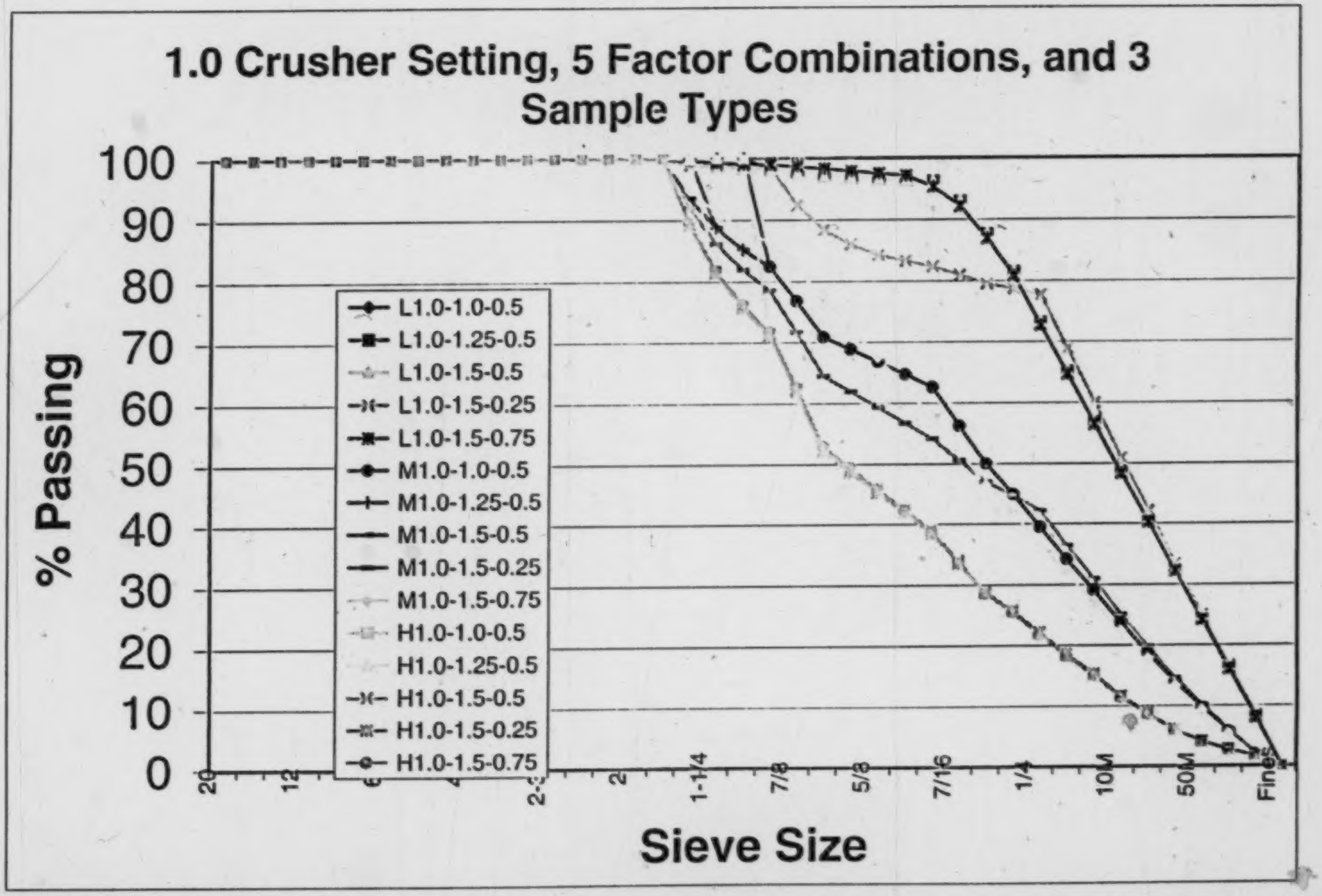
APPENDIX M

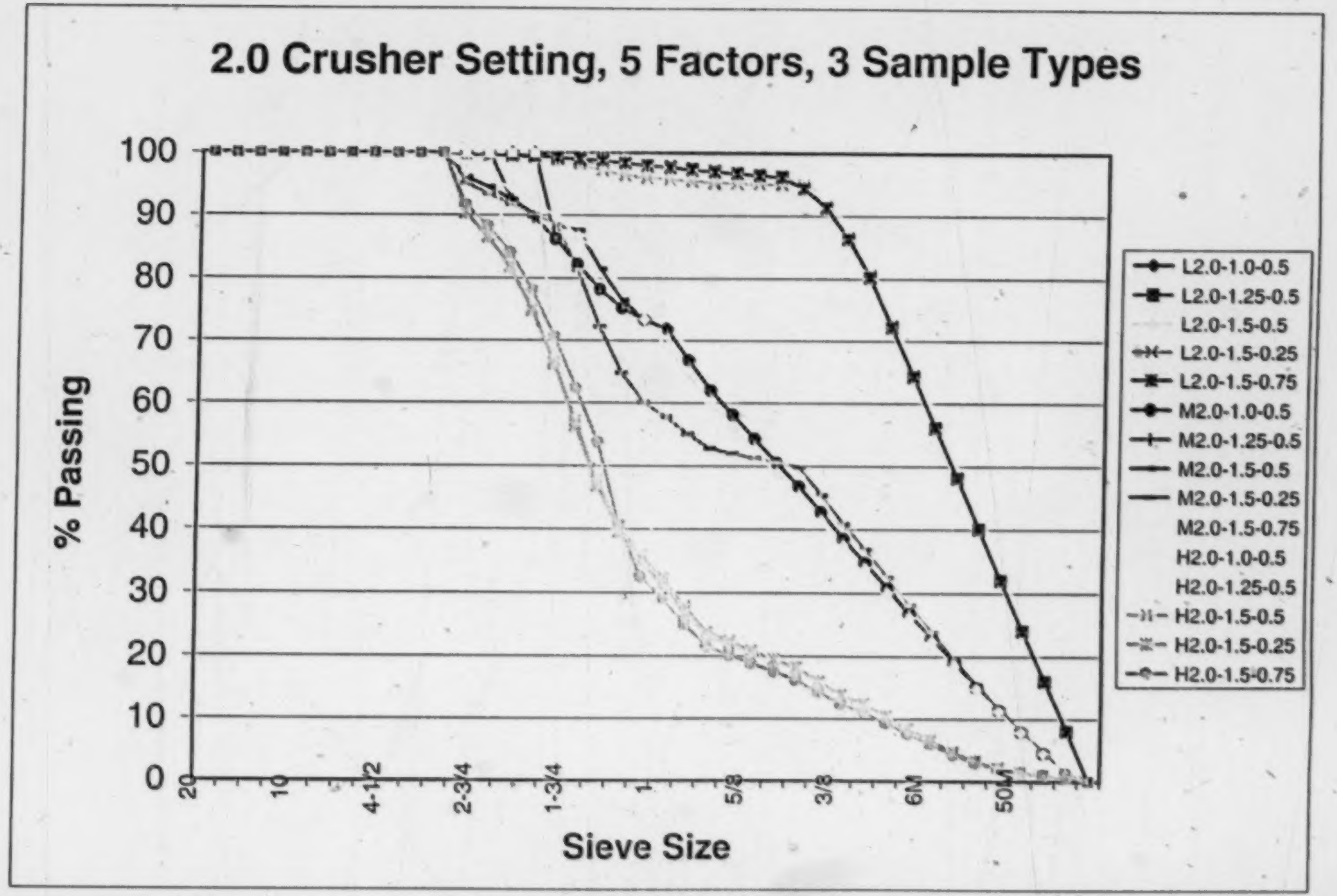


Low, Medium, High Inputs and An Actual Gradation



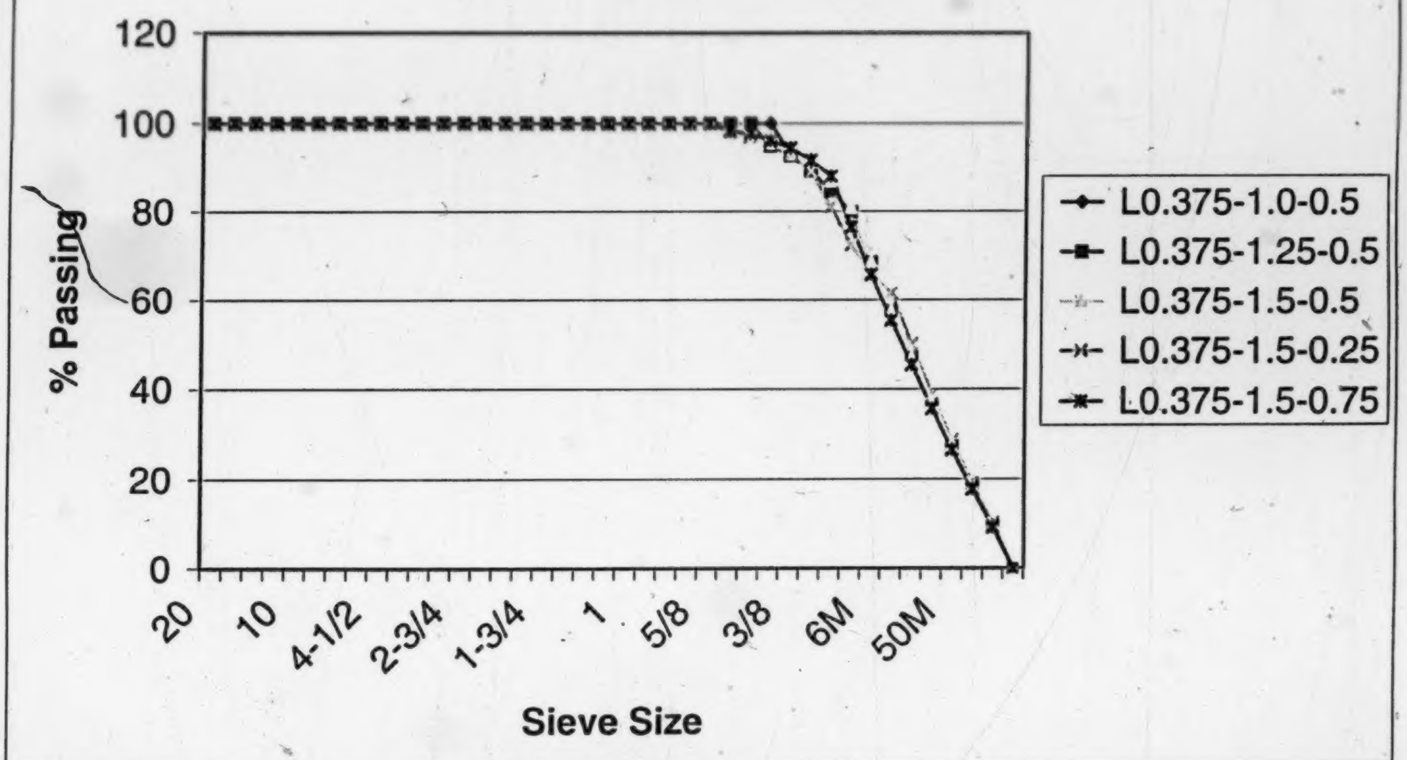
0.375 Crusher Setting, All 5 Factor Combinations, and 3 Sample Types

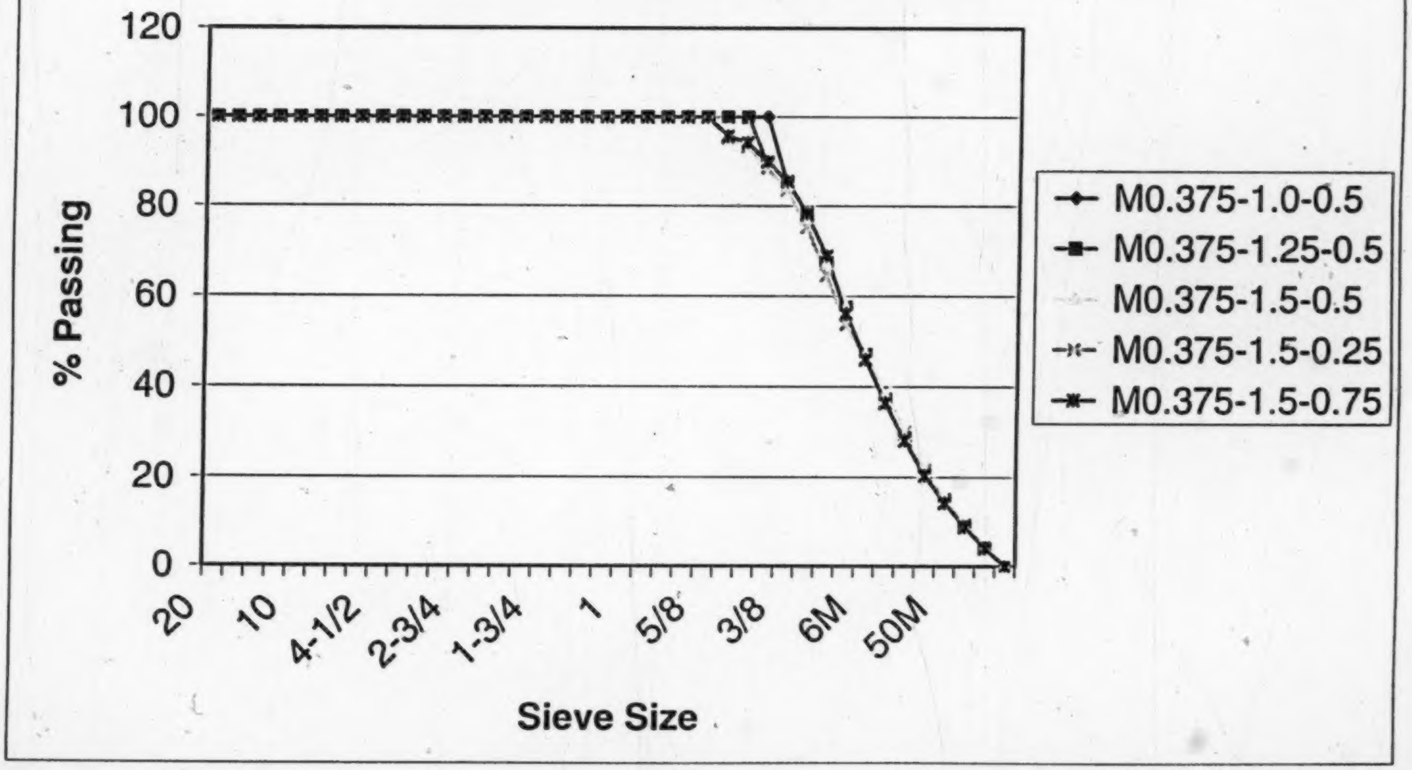

1.0 Crusher Setting, 5 Factor Combinations, and 3 Sample Types
% Passing
100
90
80
70
60
50
40
30
20
10
0
Sieve Size
20
12
6
4
2-3
2
1-1/4
7/8
5/8
7/16
1/4
10M
50M
Fines
L1.0-1.0-0.5
L1.0-1.25-0.5
L1.0-1.5-0.5
L1.0-1.5-0.25
L1.0-1.5-0.75
M1.0-1.0-0.5
M1.0-1.25-0.5
M1.0-1.5-0.5
M1.0-1.5-0.25
M1.0-1.5-0.75
H1.0-1.0-0.5
H1.0-1.25-0.5
H1.0-1.5-0.5
H1.0-1.5-0.25
H1.0-1.5-0.75



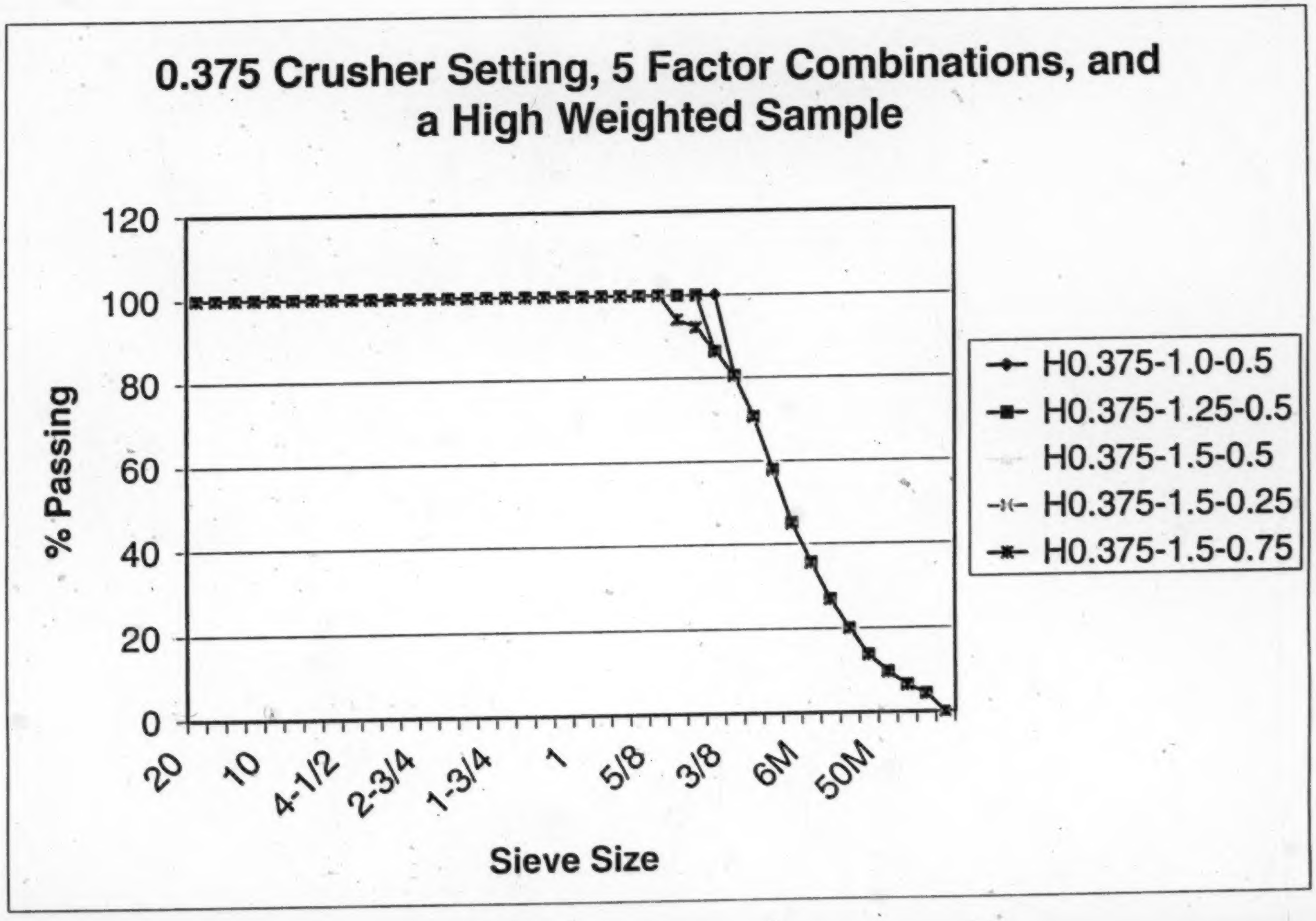
2.0 Crusher Setting, 5 Factors, 3 Sample Types

0.375 Crusher Setting, 5 Factor Combinations, and a Low Weighted Sample

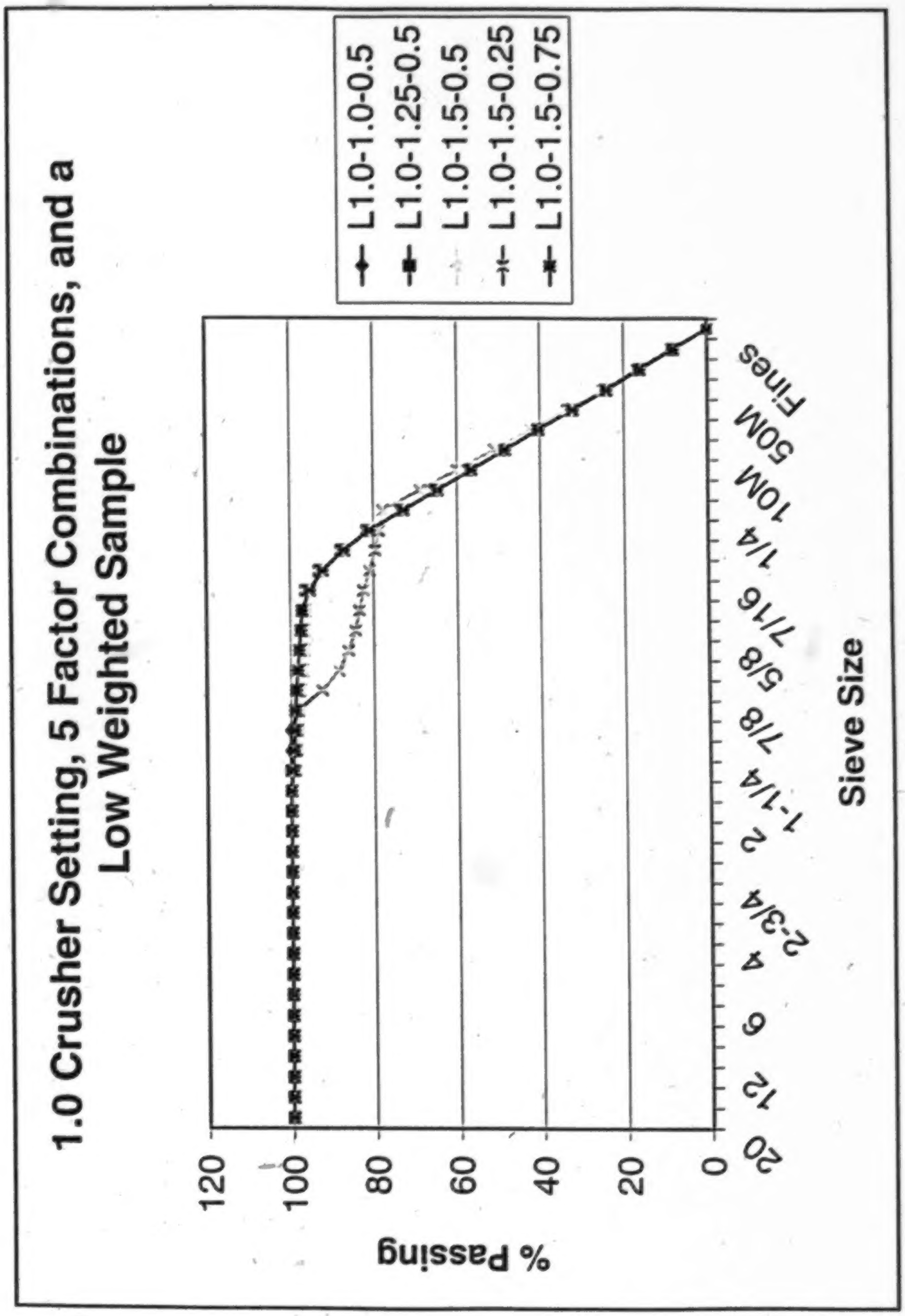


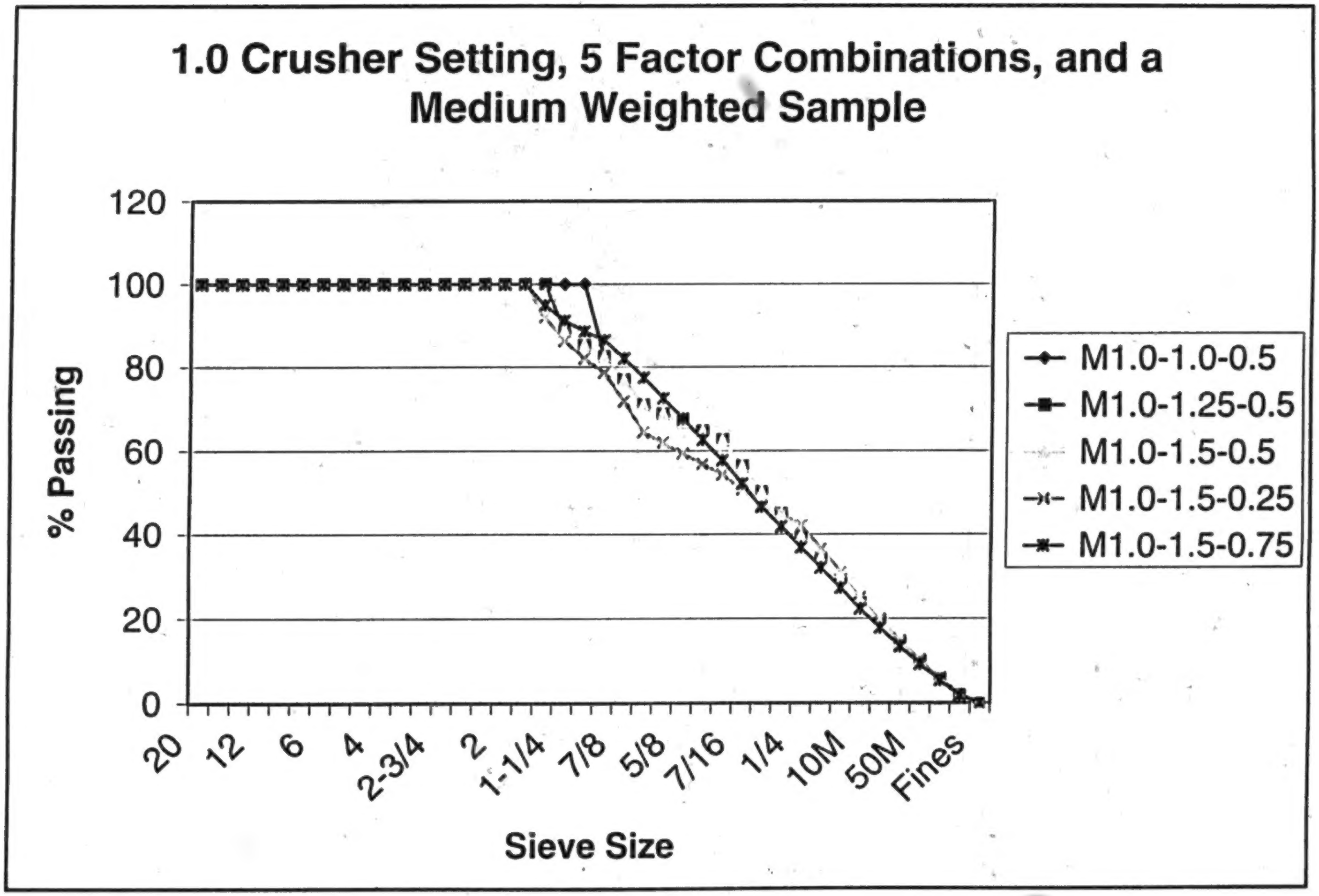


0.375 Crusher Setting, 5 Factor Combinations, and a Medium Weighted Sample

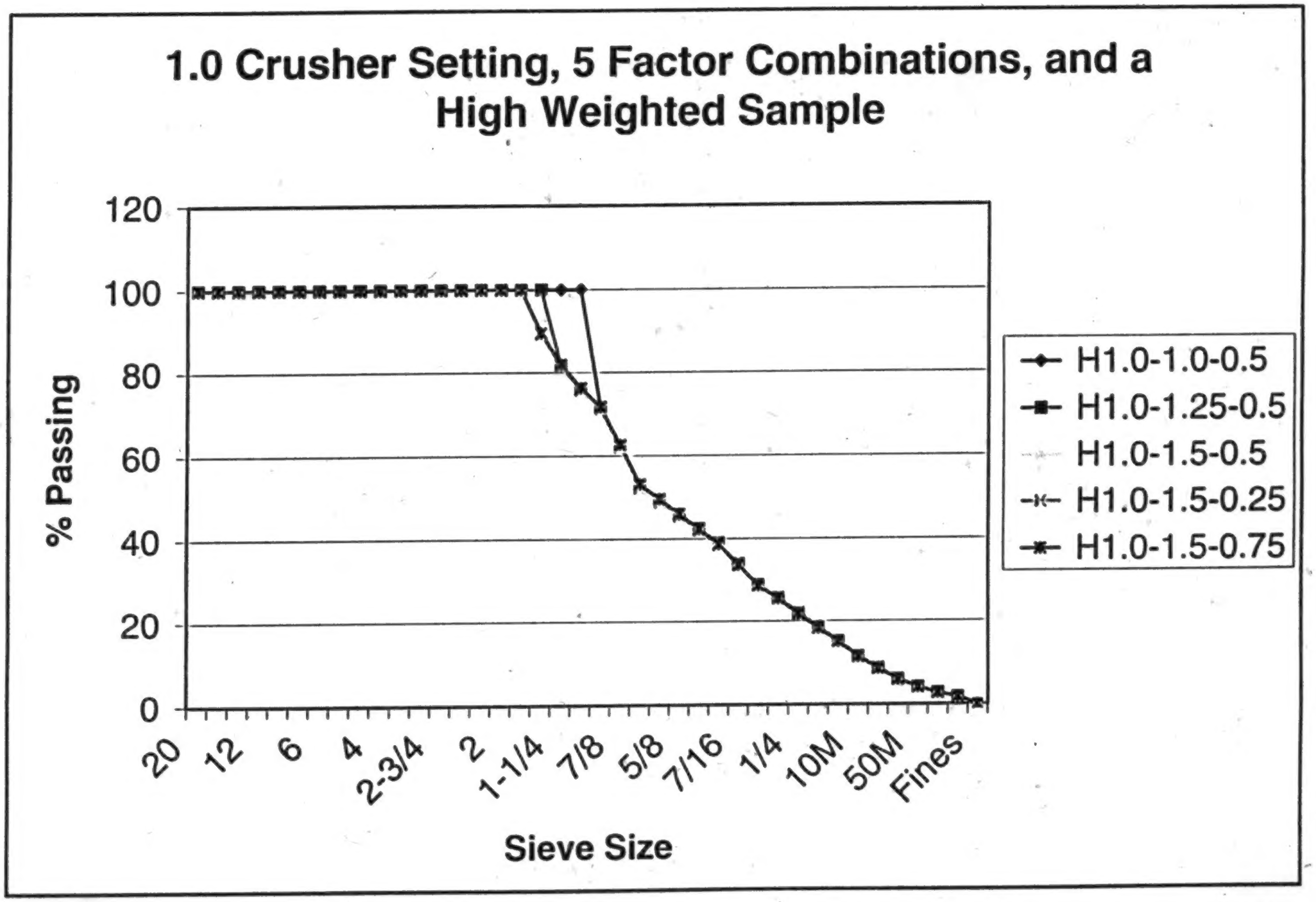


0.375 Crusher Setting, 5 Factor Combinations, and a High Weighted Sample

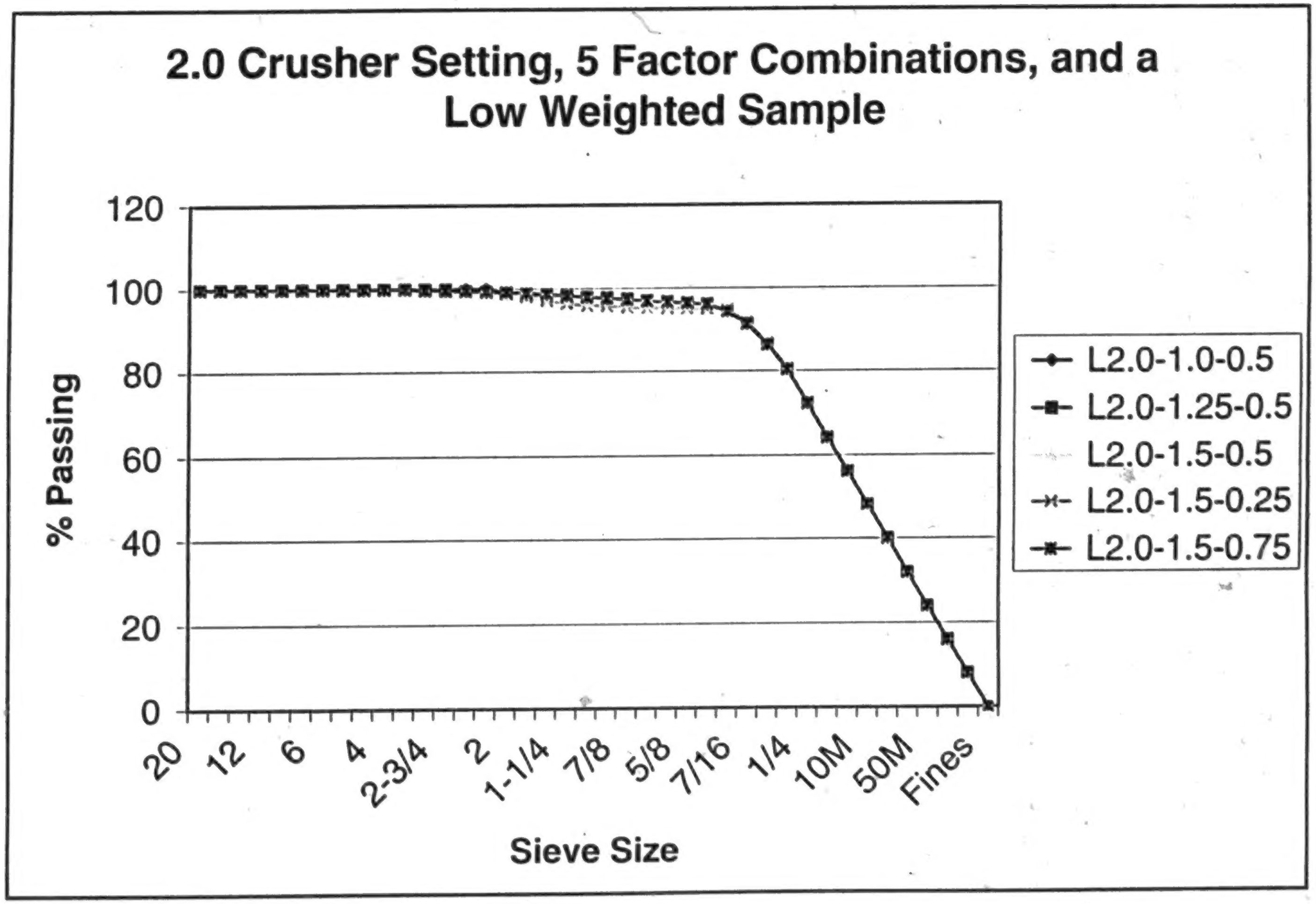

1.0 Crusher Setting, 5 Factor Combinations, and a Low Weighted Sample
% Passing
120
100
80
60
40
20
0
Sieve Size
20
12
6
4
2-3/4
2
1-1/4
7/8
5/8
7/16
1/4
10M
50M
Fines
L1.0-1.0-0.5
L1.0-1.25-0.5
L1.0-1.5-0.5
L1.0-1.5-0.25
L1.0-1.5-0.75



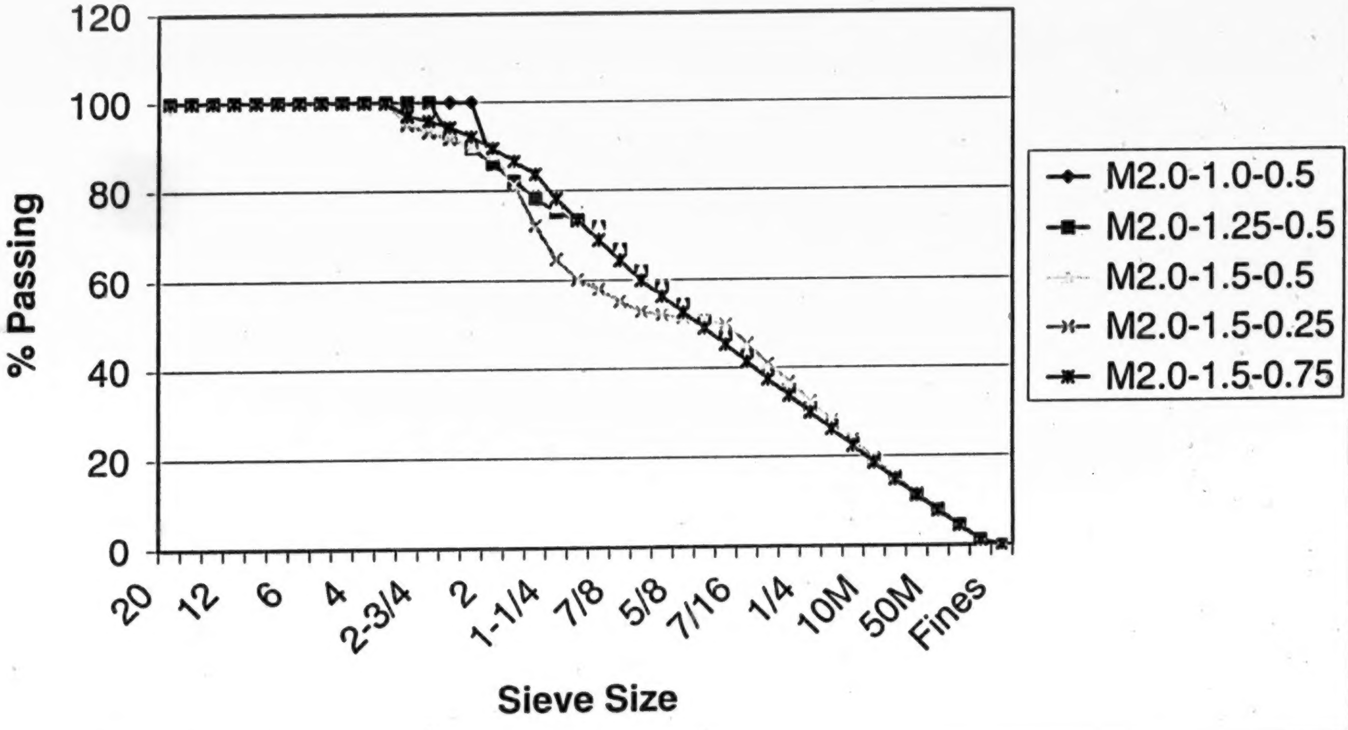
1.0 Crusher Setting, 5 Factor Combinations, and a Medium Weighted Sample

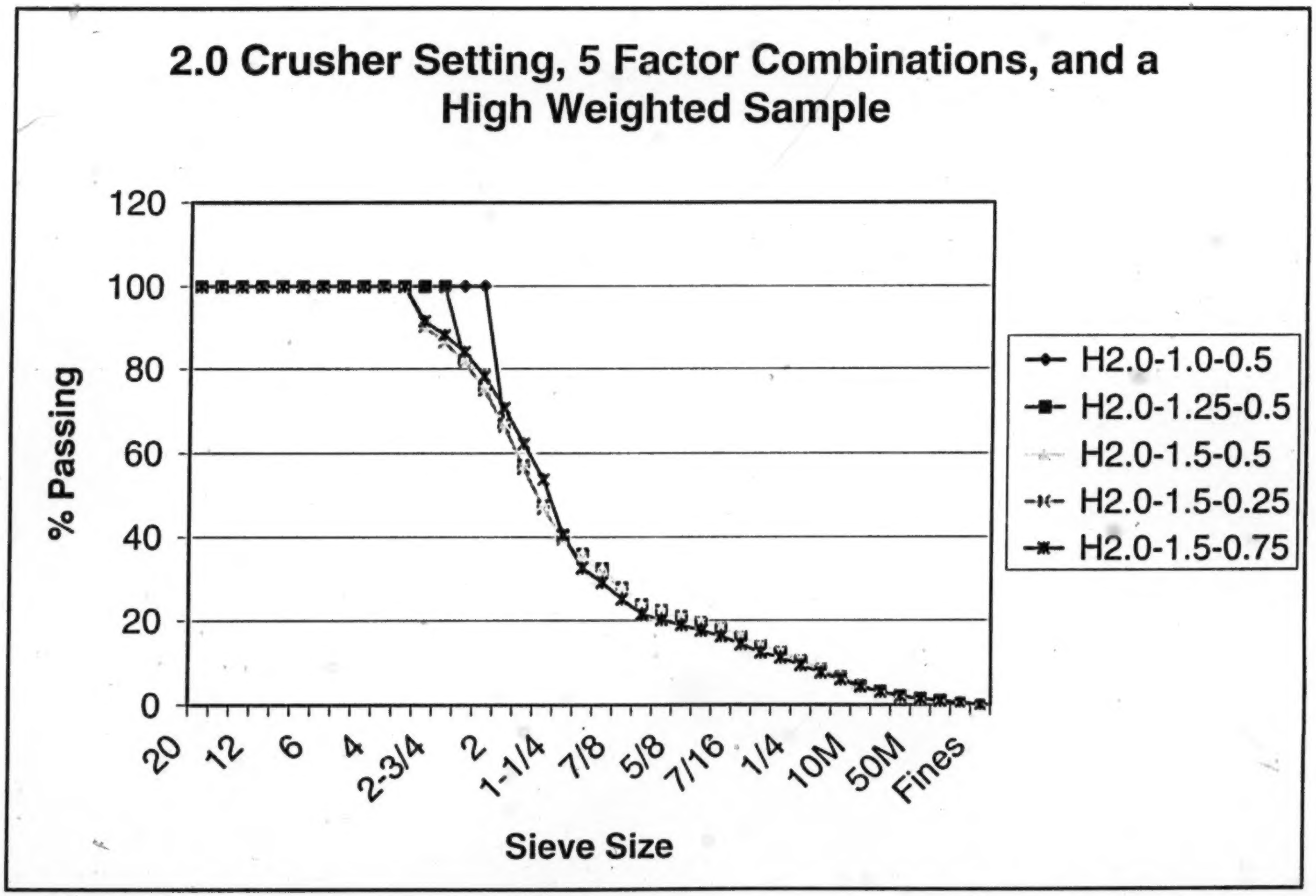


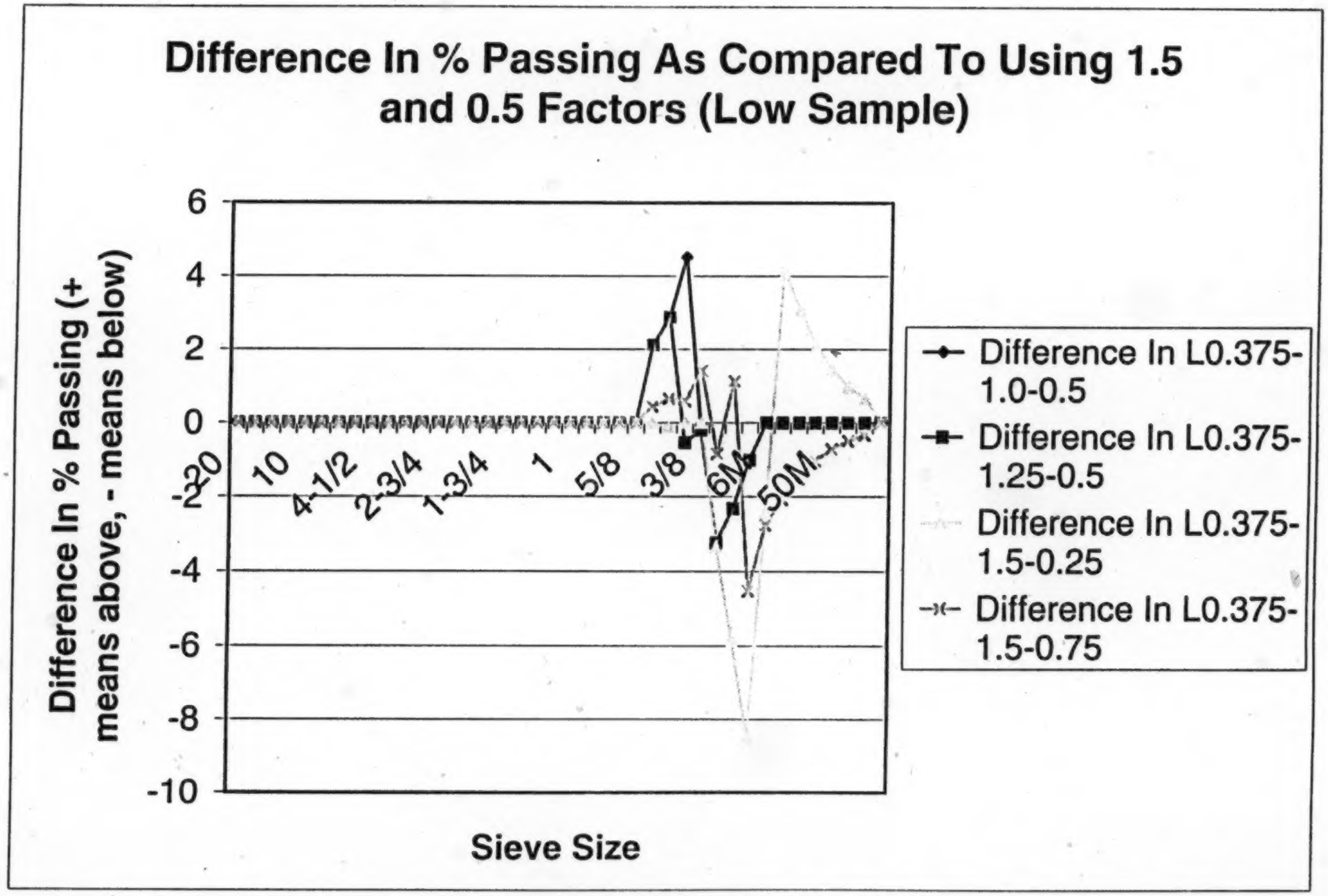
1.0 Crusher Setting, 5 Factor Combinations, and a High Weighted Sample


2.0 Crusher Setting, 5 Factor Combinations, and a Low Weighted Sample
% Passing
120
100
80
60
40
20
0
20
12
6
4
2-3/4
2
1-1/4
7/8
5/8
7/16
1/4
10M
50M
Fines
Sieve Size
L2.0-1.0-0.5
L2.0-1.25-0.5
L2.0-1.5-0.5
L2.0-1.5-0.25
L2.0-1.5-0.75

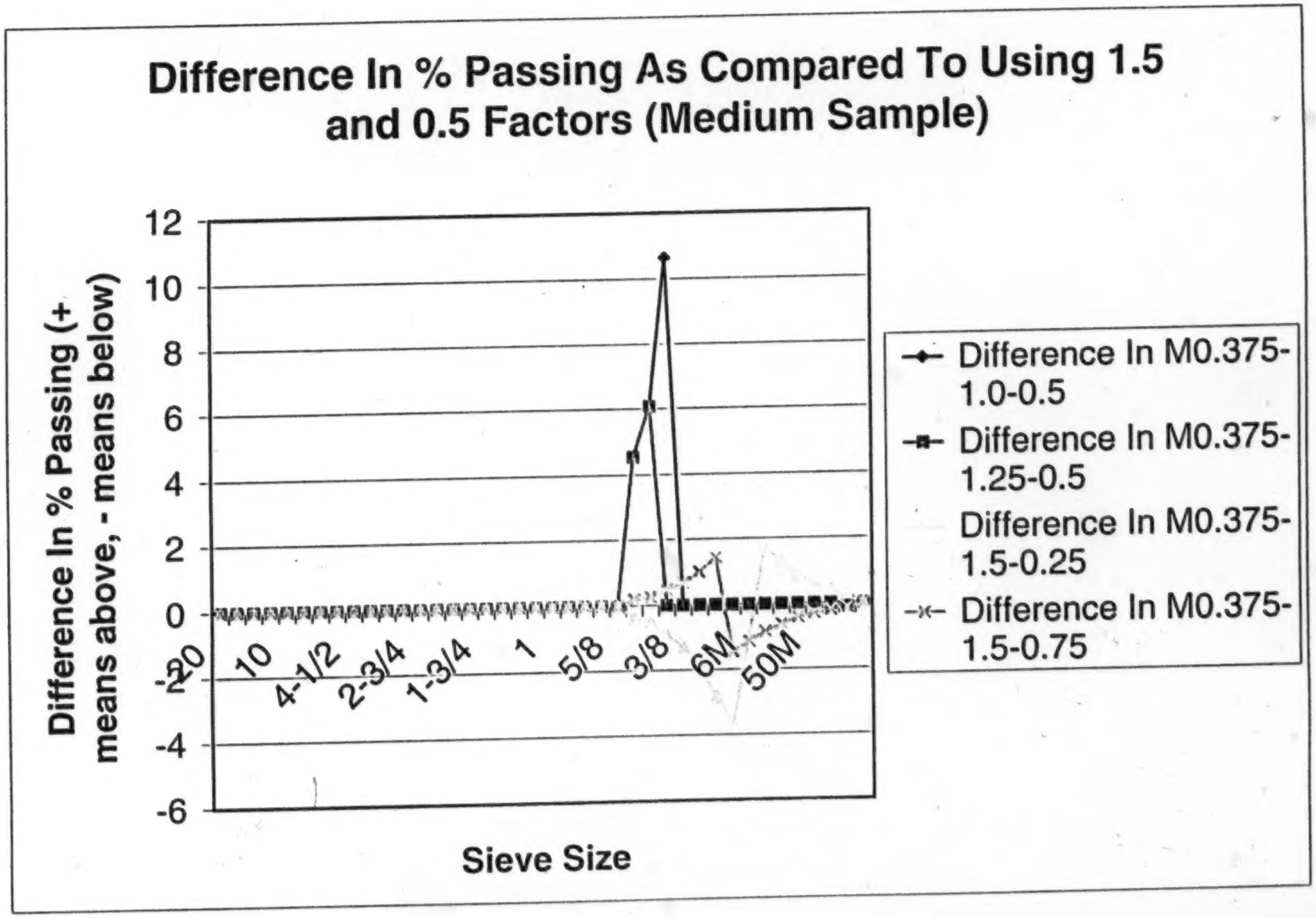
2.0 Crusher Setting, 5 Factor Combinations, and a Medium Weighted Sample



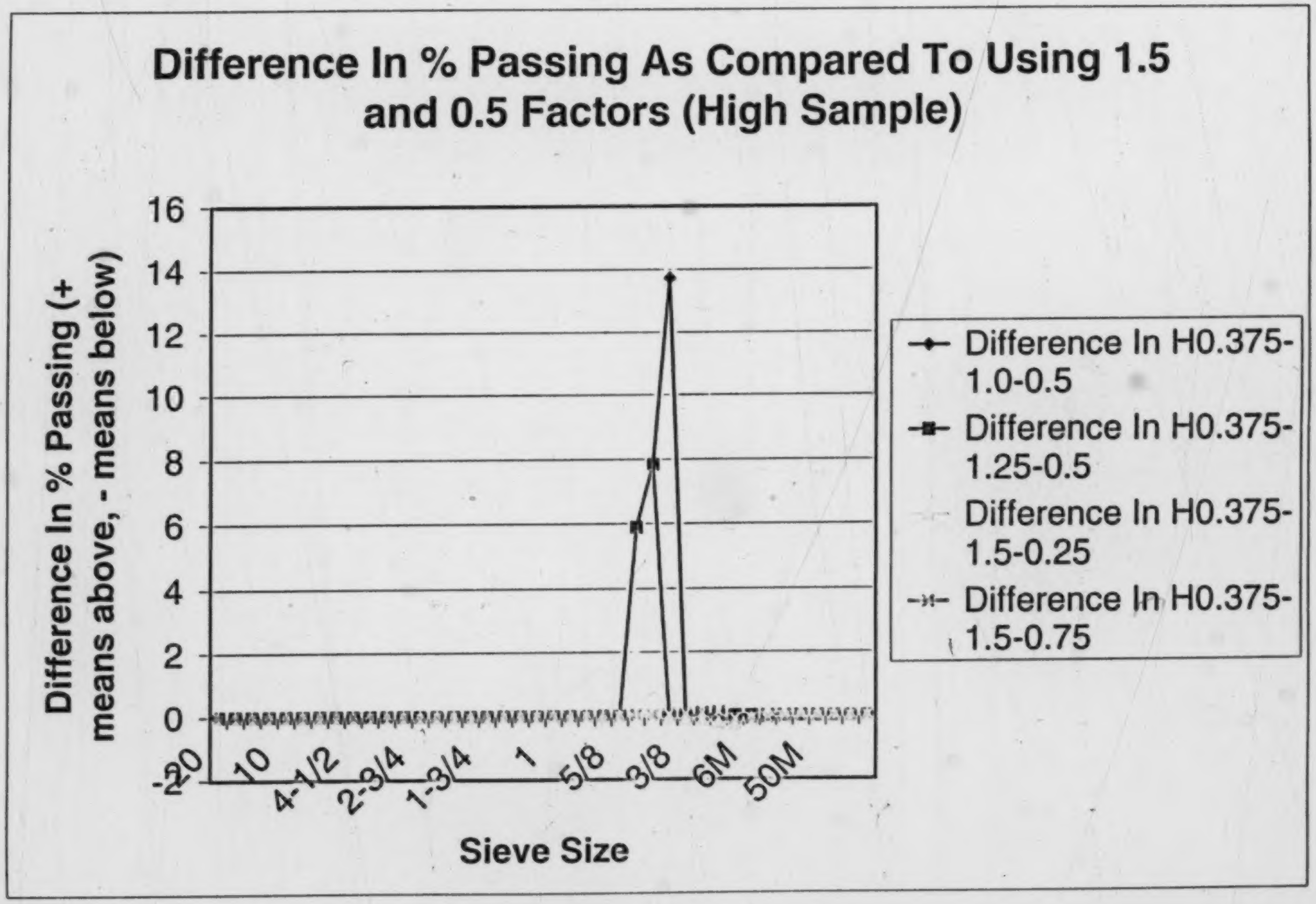

2.0 Crusher Setting, 5 Factor Combinations, and a High Weighted Sample
% Passing
120
100
80
60
40
20
0
20
12
6
4
2-3/4
2
1-1/4
7/8
5/8
7/16
1/4
10M
50M
Fines
Sieve Size
H2.0-1.0-0.5
H2.0-1.25-0.5
H2.0-1.5-0.5
H2.0-1.5-0.25
H2.0-1.5-0.75

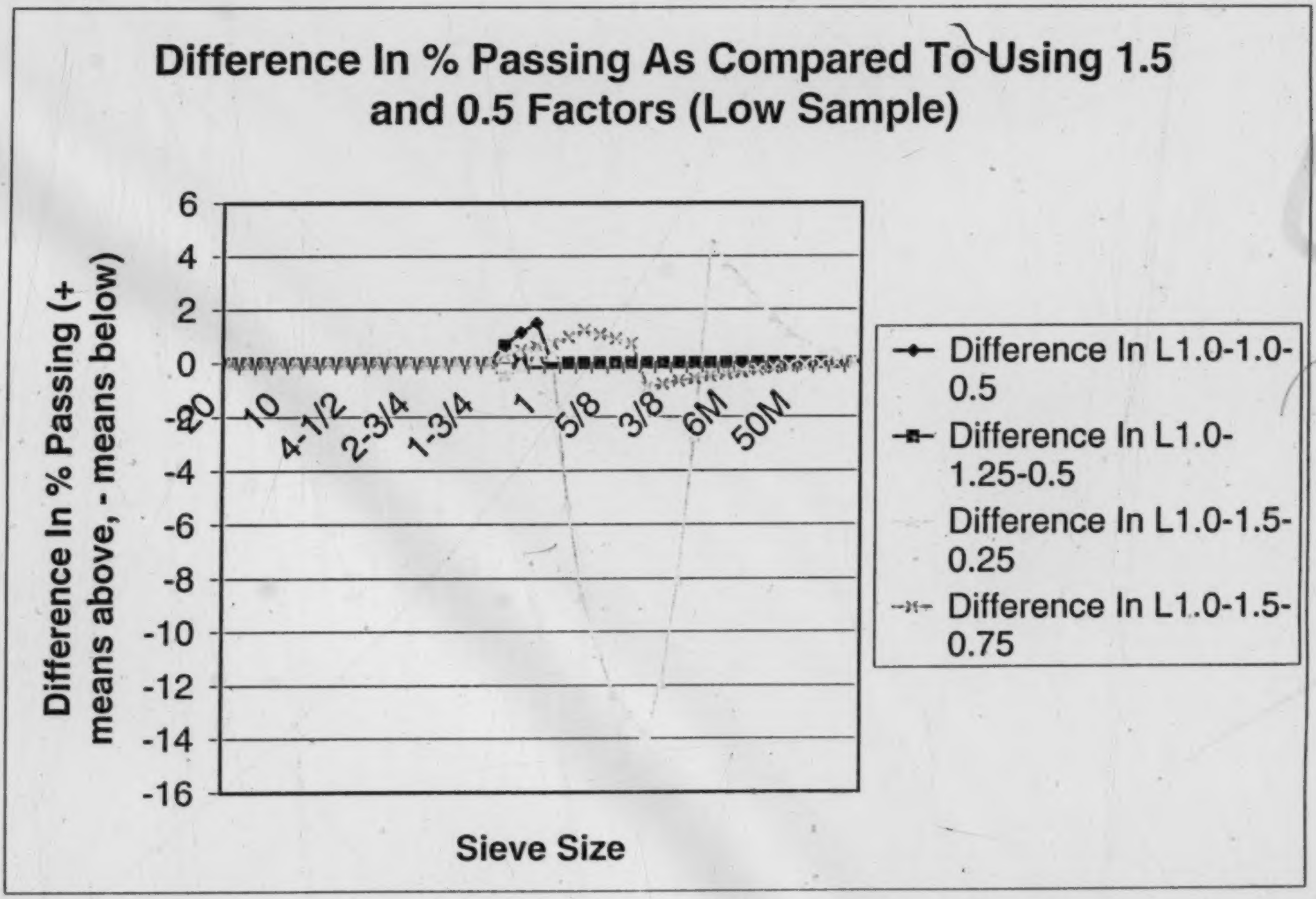


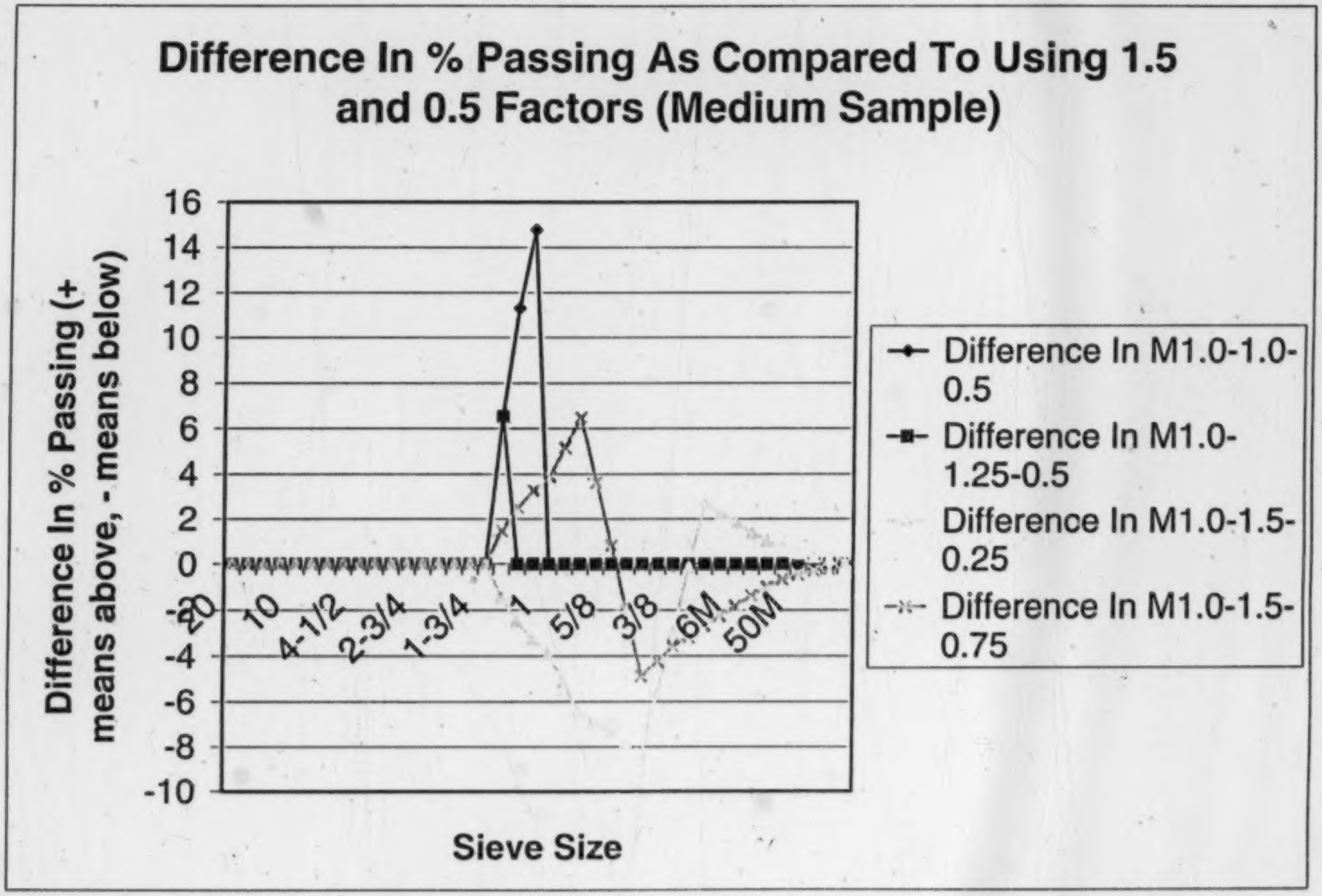
Difference In % Passing As Compared To Using 1.5 and 0.5 Factors (Low Sample)

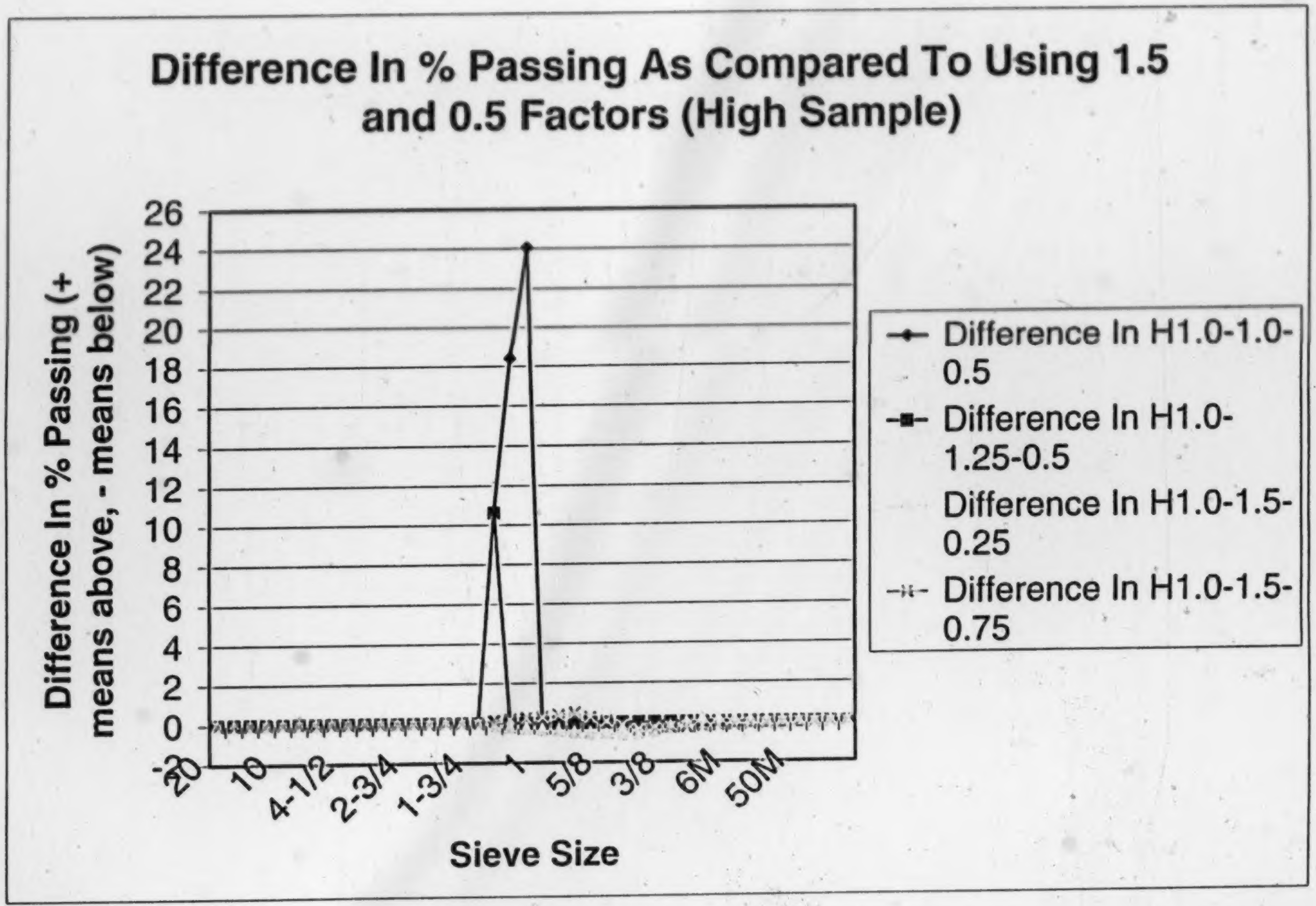


Difference In % Passing As Compared To Using 1.5 and 0.5 Factors (Medium Sample)

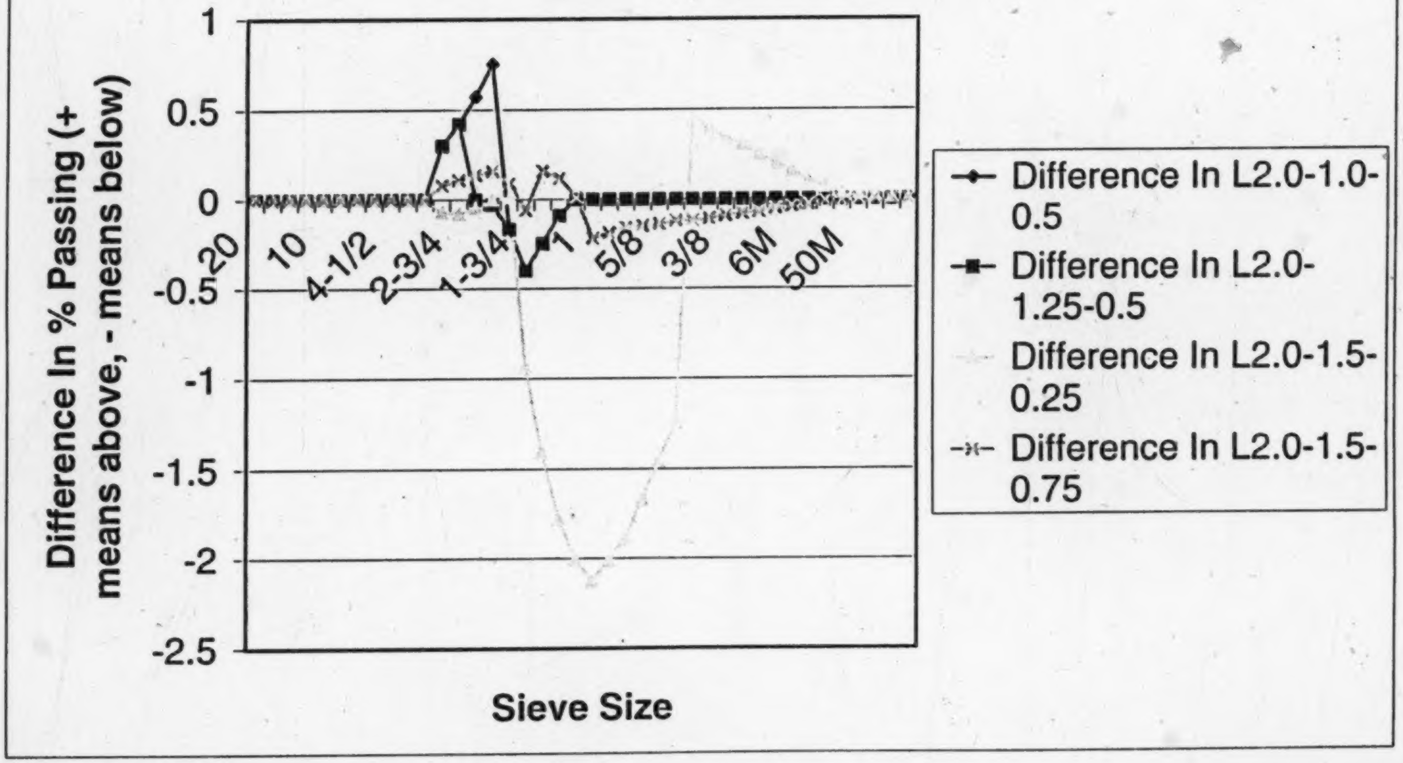

Difference In % Passing As Compared To Using 1.5 and 0.5 Factors (High Sample)
Difference In % Passing (+ means above, - means below)
16
14
12
10
8
6
4
2
0
-2
-20
10
4-1/2
2-3/4
1-3/4
1
5/8
3/8
6M
50M
Sieve Size
Difference In H0.375-1.0-0.5
Difference In H0.375-1.25-0.5
Difference In H0.375-1.5-0.25
Difference In H0.375-1.5-0.75

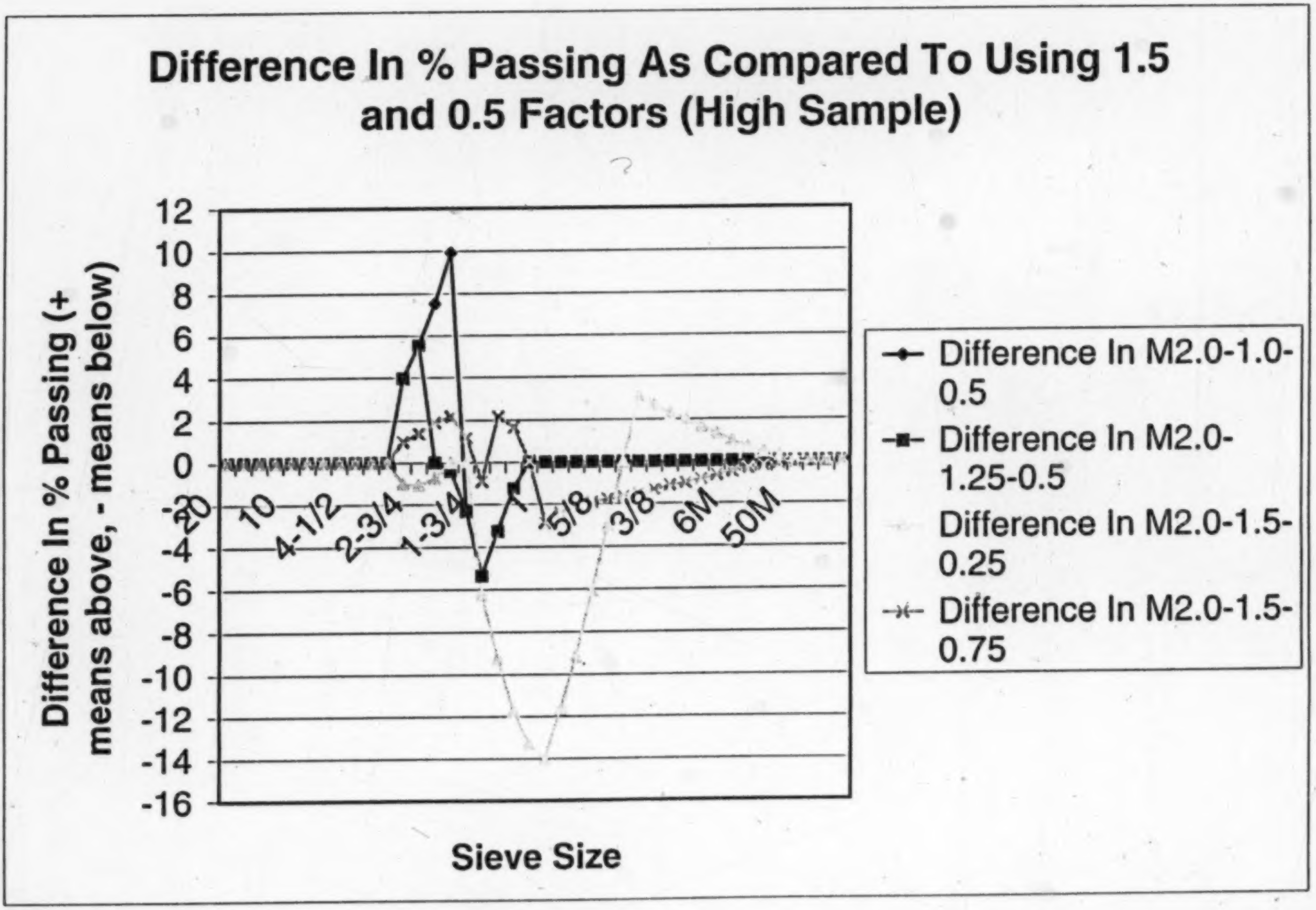

Difference In % Passing As Compared To Using 1.5 and 0.5 Factors (Low Sample)
Difference In % Passing (+ means above, - means below)
6
4
2
0
-2
-4
-6
-8
-10
-12
-14
-16
20
10
4-1/2
2-3/4
1-3/4
1
5/8
3/8
6M
50M
Sieve Size
Difference In L1.0-1.0-0.5
Difference In L1.0-1.25-0.5
Difference In L1.0-1.5-0.25
Difference In L1.0-1.5-0.75


Difference In % Passing As Compared To Using 1.5 and 0.5 Factors (Medium Sample)
Difference In % Passing (+ means above, - means below)
16
14
12
10
8
6
4
2
0
-2
-4
-6
-8
-10
20
10
4-1/2
2-3/4
1-3/4
1
5/8
3/8
6M
50M
Sieve Size
Difference In M1.0-1.0-0.5
Difference In M1.0-1.25-0.5
Difference In M1.0-1.5-0.25
Difference In M1.0-1.5-0.75

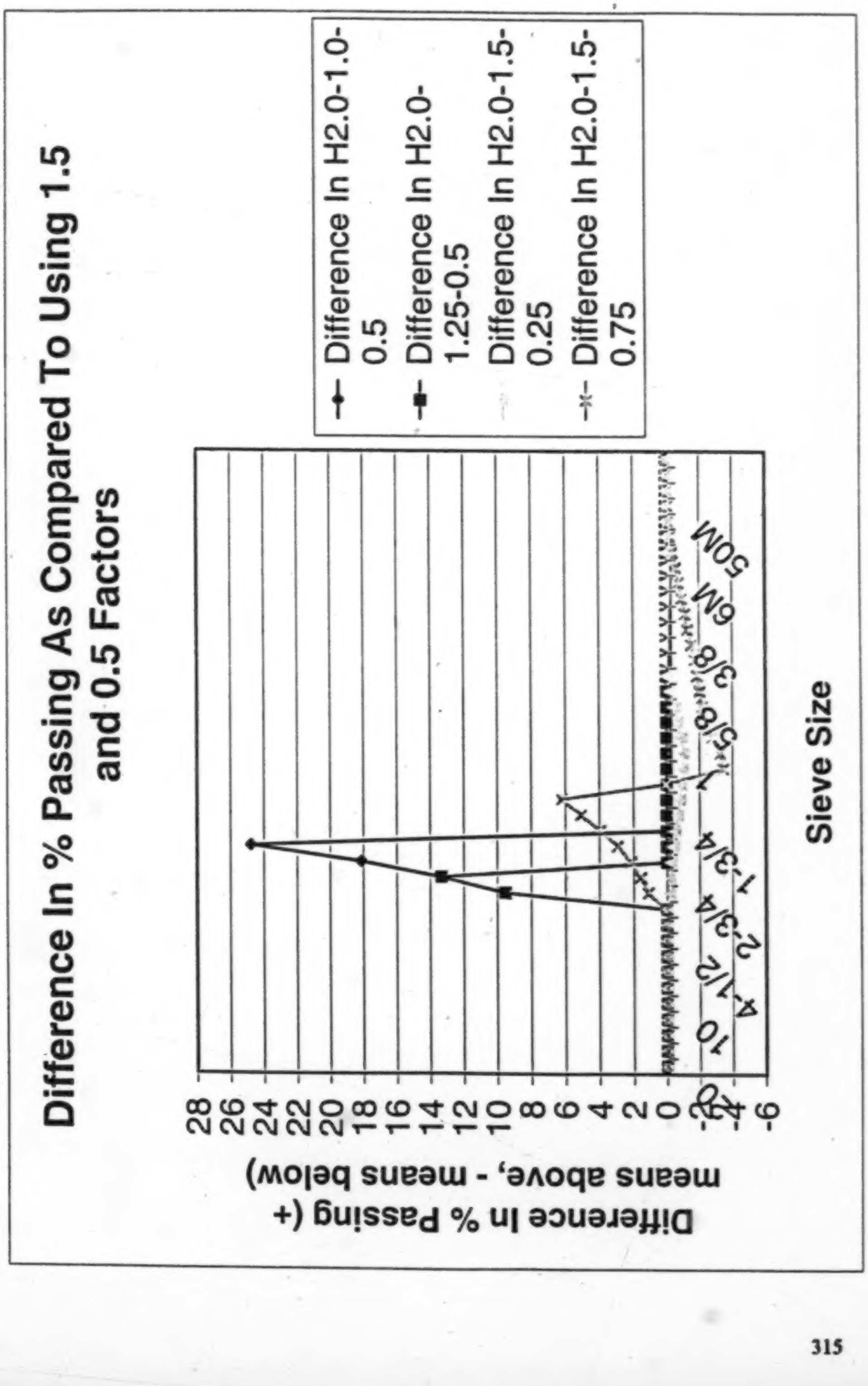

Difference In % Passing As Compared To Using 1.5 and 0.5 Factors (High Sample)
Difference In % Passing (+ means above, - means below)
26
24
22
20
18
16
14
12
10
8
6
4
2
0
-2
20
10
4-1/2
2-3/4
1-3/4
1
5/8
3/8
6M
50M
Sieve Size
Difference In H1.0-1.0-0.5
Difference In H1.0-1.25-0.5
Difference In H1.0-1.5-0.25
Difference In H1.0-1.5-0.75

Difference In % Passing As Compared To Using 1.5 and 0.5 Factors (High Sample)

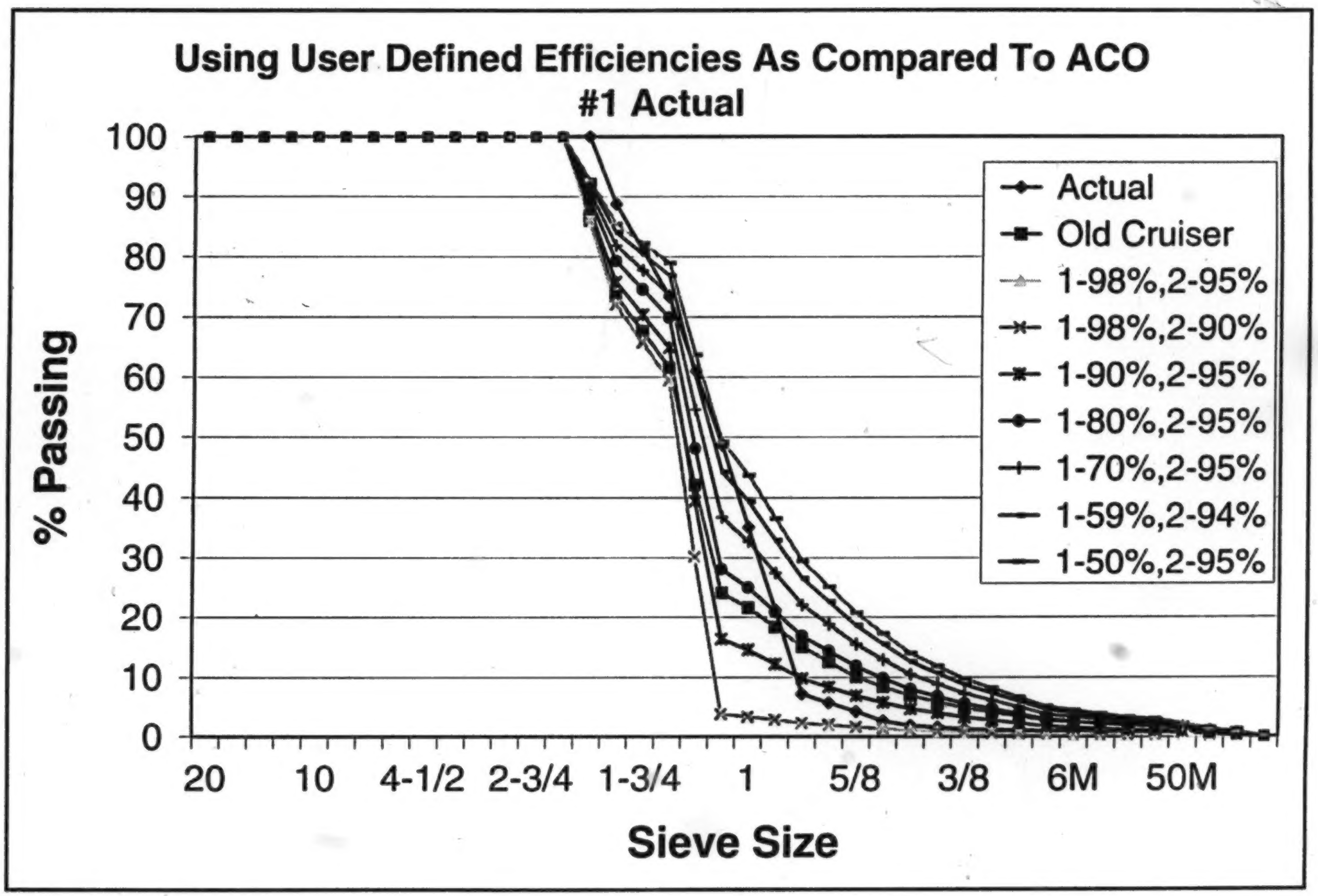




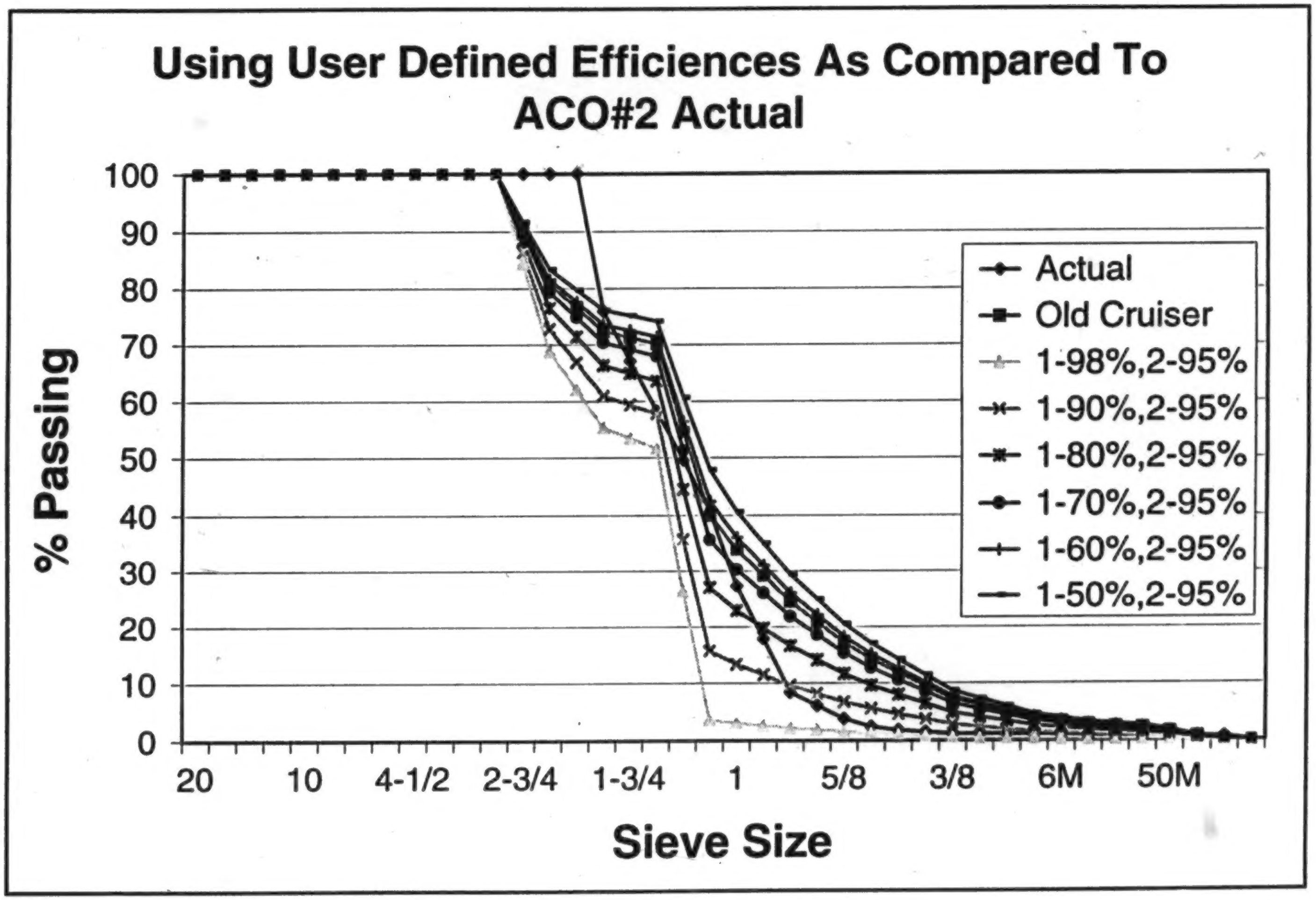
Difference In % Passing As Compared To Using 1.5 and 0.5 Factors (High Sample)


Difference In % Passing As Compared To Using 1.5 and 0.5 Factors

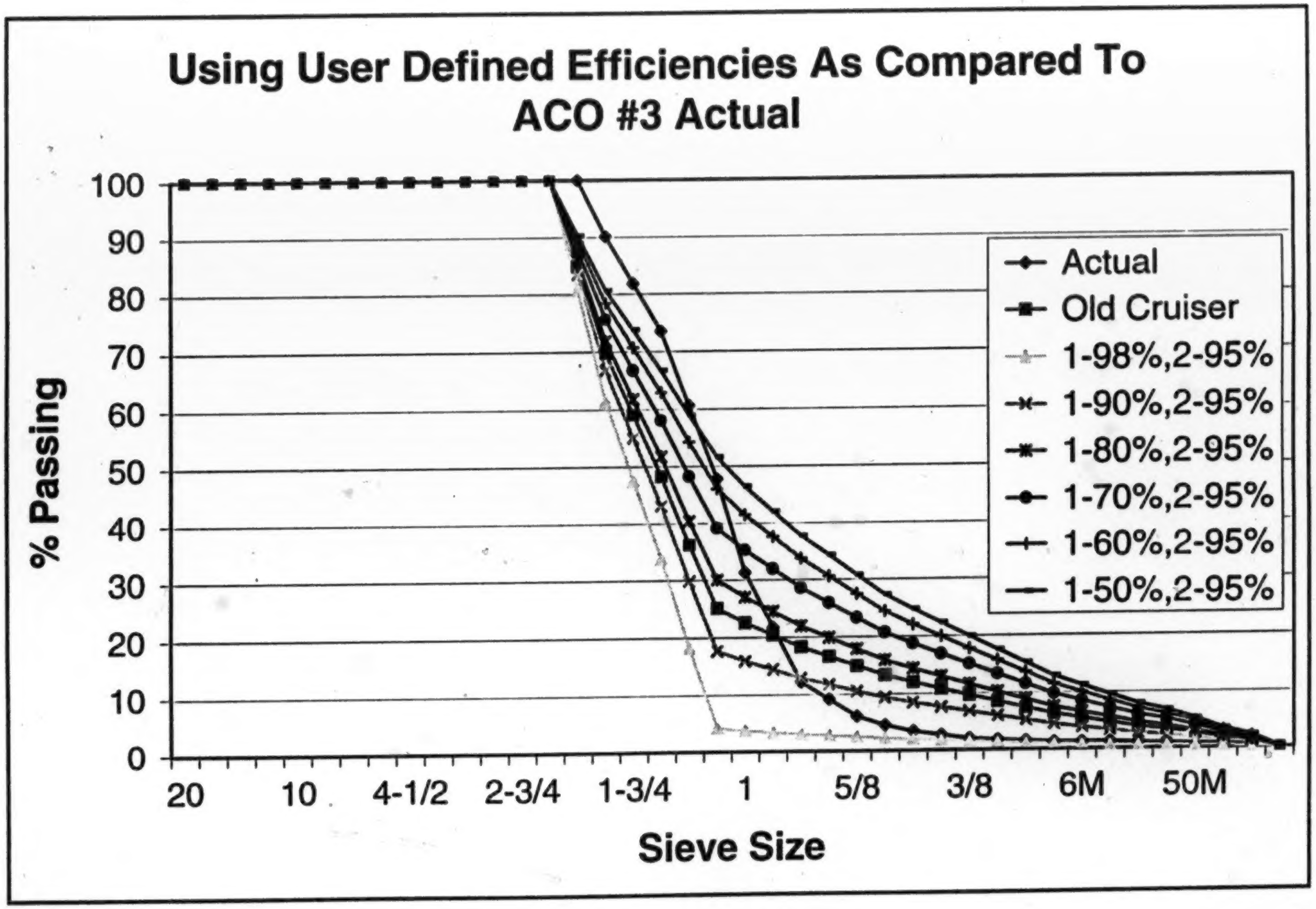
APPENDIX N

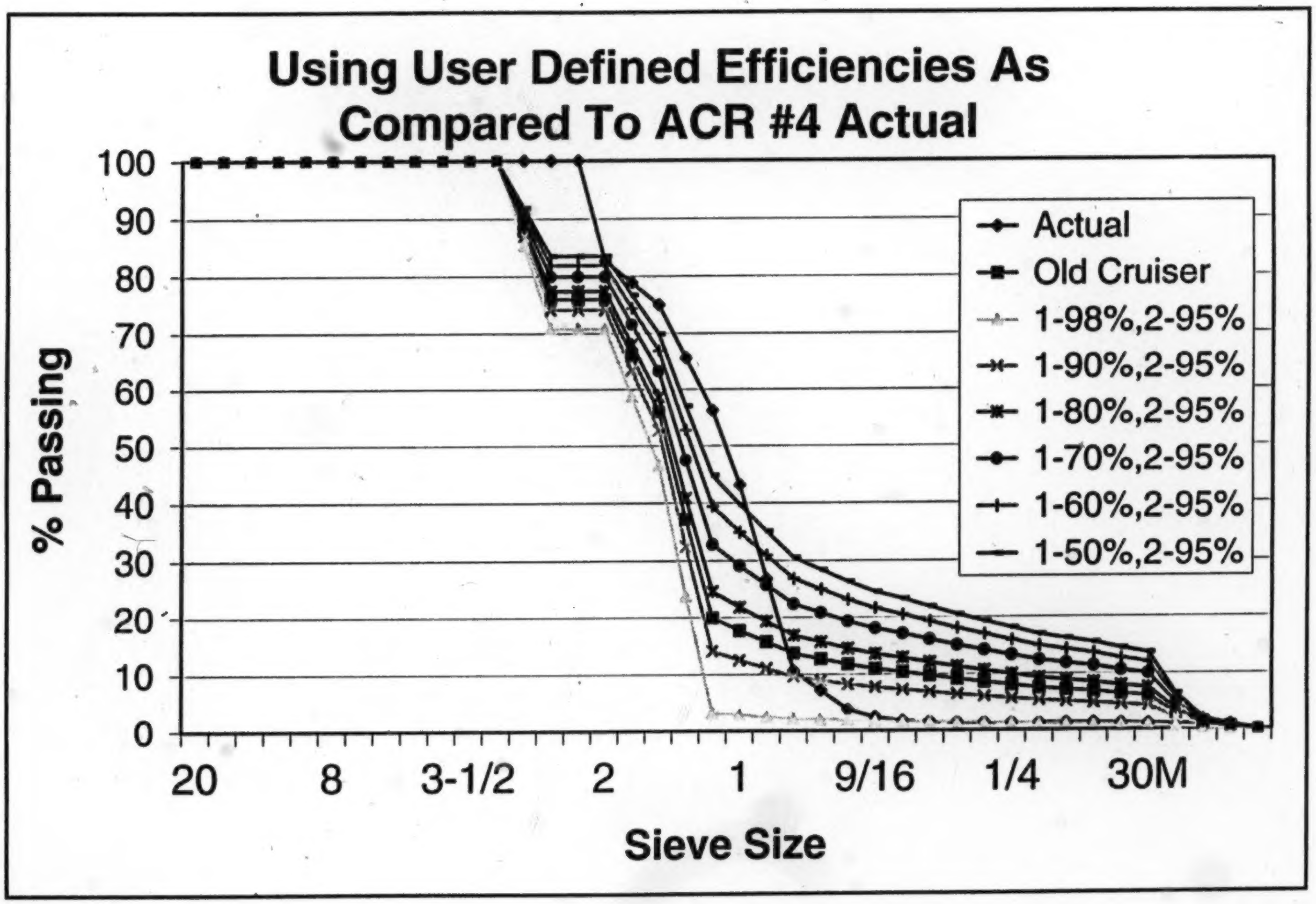


Using User Defined Efficiencies As Compared To ACO
#1 Actual



Using User Defined Efficiencies As Compared To ACO#2 Actual


Using User Defined Efficiencies As Compared To ACO #3 Actual
% Passing
100
90
80
70
60
50
40
30
20
10
0
20
10
4-1/2
2-3/4
1-3/4
1
5/8
3/8
6M
50M
Sieve Size
Actual
Old Cruiser
1-98%,2-95%
1-90%,2-95%
1-80%,2-95%
1-70%,2-95%
1-60%,2-95%
1-50%,2-95%



Using User Defined Efficiencies As Compared To ACR #4 Actual

APPENDIX O

```
Option Explicit

Private Declare Function crush Lib "numrec.dll" (nstin As Single, ByVal sset As Single, nsout A
s Single, ByVal ntype As Single, ByVal itype As Single, SIZES As Single, ByVal nsieve As Single
) As Integer




'Description: generates random #s, utilizes the "CRUSH" routine and creates NN input and output
 for training purposes
'Input: # of different random combinations to be sent to the "CRUSH" routine for all ten crushe
r settings
'Output: NN input and NN output for training purposes
'Side Effects: the nstin and nsout arrays are passed to other functions and change for each set
 of different random numbers
Private Sub cmdrun_Click()

' Declare Variables
    Dim nstin(0 To 41) As Single
    Dim sset As Single
    Dim nsout(0 To 41) As Single
    Dim ntype As Single
    Dim itype As Single
    Dim nsizes(0 To 41) As Single
    Dim nsieve As Single
    Dim result As Integer
    Dim i As Integer
    Dim j As Integer
    Dim k As Integer
    Dim l As Integer
    Dim Sum As Single
    Dim MyDb As Database
    Dim SetArray(0 To 10) As Single

' Initialize Parameters (leave ntype as 3(ie cone crusher) and itype as 1)
    'sset = Val(txtset.Text)
    ntype = 3
    itype = 1
    nsieve = 39

' Initialize the Sieve Size Array

    nsizes(1) = 20
    nsizes(2) = 16
    nsizes(3) = 14
    nsizes(4) = 12
    nsizes(5) = 10
    nsizes(6) = 8
    nsizes(7) = 6
    nsizes(8) = 5
    nsizes(9) = 4.5
    nsizes(10) = 4
    nsizes(11) = 3.5
    nsizes(12) = 3
    nsizes(13) = 2.75
    nsizes(14) = 2.5
    nsizes(15) = 2.25
    nsizes(16) = 2
    nsizes(17) = 1.75
    nsizes(18) = 1.5
    nsizes(19) = 1.25
    nsizes(20) = 1.11111
    nsizes(21) = 1
    nsizes(22) = 0.875
    nsizes(23) = 0.75
    nsizes(24) = 0.6875
    nsizes(25) = 0.625
    nsizes(26) = 0.5625
    nsizes(27) = 0.5
    nsizes(28) = 0.4375
    nsizes(29) = 0.375
    nsizes(30) = 0.3125
    nsizes(31) = 0.25
    'all the imperial "M sizes" come after this line
    nsizes(32) = 0.187
    nsizes(33) = 0.132
    nsizes(34) = 0.0787
    nsizes(35) = 0.0469
    nsizes(36) = 0.0234
    nsizes(37) = 0.0117
    nsizes(38) = 0.0059
    nsizes(39) = 0.0029

'set the number of data sets to be created
'k = Val(txtnumbset.Text)
'For l = 1 To k

'set the database for data storage
Set MyDb = DBEngine.Workspaces(0).OpenDatabase("c:\civil96\civ603\project\nndata.mdb")

'initializing the crusher setting array with the 10 settings available in the "CRUSH" routine

    SetArray(1) = 2
    SetArray(2) = 1.75
    SetArray(3) = 1.5
    SetArray(4) = 1.25
    SetArray(5) = 1
    SetArray(6) = 0.875
    SetArray(7) = 0.75
    SetArray(8) = 0.5
    SetArray(9) = 0.4375
    SetArray(10) = 0.375

'set the number of data sets to be created
'set k to 500 when actual data is to be obtained (for a total of 5000 input and 5000 output gra
dations)
k = 500
For l = 1 To k

    'obtaining random numbers for %retained for each sieve
        For i = 1 To 40
            nstin(i) = 1000 * Rnd()
        Next

    'data is sent to a temporary table to sort the data
    DataTemp nstin, nsizes, l, MyDb

'translate the %retained to weight retained on each sieve
    nstin(1) = nstin(1)
    nstin(2) = nstin(2) - nstin(1)
    nstin(3) = nstin(3) - nstin(2) - nstin(1)
    nstin(4) = nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(5) = nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(6) = nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(7) = nstin(7) - nstin(6) - nstinn(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(8) = nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nst
in(1)
    nstin(9) = nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nst
in(2) - nstin(1)
    nstin(10) = nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - n
stin(3) - nstin(2) - nstin(1)
    nstin(11) = nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) -
nstin(4) - nstin(3) - nstin(2) - nstin(1)
```

```
    nstin(12) = nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) -
 nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(13) = nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7)
- nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(14) = nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8)
 - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(15) = nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9
) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(16) = nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(1
0) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - ns
tin(1)
    nstin(17) = nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(1
1) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - n
stin(2) - nstin(1)
    nstin(18) = nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(1
2) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) -
nstin(3) - nstin(2) - nstin(1)
    nstin(19) = nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(1
3) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) -
 nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(20) = nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(1
4) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6)
- nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(21) = nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(1
5) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7)
 - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(22) = nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(1
6) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8
) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(23) = nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(1
7) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(
9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(24) = nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(1
8) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(
10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - n
stin(1)
    nstin(25) = nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(1
9) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(
11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) -
nstin(2) - nstin(1)
    nstin(26) = nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(2
0) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(
12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) -
 nstin(3) - nstin(2) - nstin(1)
    nstin(27) = nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(2
1) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(
13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5)
- nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(28) = nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(2
2) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(
14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6)
 - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(29) = nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(2
3) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(
15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7
) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(30) = nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(2
4) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(
16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(
8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(31) = nstin(31) - nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(2
5) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(
17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin
(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(32) = nstin(32) - nstin(31) - nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(2
6) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(
18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin
(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) -
```

```
nnInput - 4

nstin(1)
    nstin(33) = nstin(33) - nstin(32) - nstin(31) - nstin(30) - nstin(29) - nstin(28) - nstin(2
7) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(
19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin
(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) -
 nstin(2) - nstin(1)
    nstin(34) = nstin(34) - nstin(33) - nstin(32) - nstin(31) - nstin(30) - nstin(29) - nstin(2
8) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(
20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin
(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4)
- nstin(3) - nstin(2) - nstin(1)
    nstin(35) = nstin(35) - nstin(34) - nstin(33) - nstin(32) - nstin(31) - nstin(30) - nstin(2
9) - nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(
21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin
(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5)
 - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(36) = nstin(36) - nstin(35) - nstin(34) - nstin(33) - nstin(32) - nstin(31) - nstin(3
0) - nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(
22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin
(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6
) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(37) = nstin(37) - nstin(36) - nstin(35) - nstin(34) - nstin(33) - nstin(32) - nstin(3
1) - nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(
23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin
(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(
7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(38) = nstin(38) - nstin(37) - nstin(36) - nstin(35) - nstin(34) - nstin(33) - nstin(3
2) - nstin(31) - nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(
24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin
(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin
(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(39) = nstin(39) - nstin(38) - nstin(37) - nstin(36) - nstin(35) - nstin(34) - nstin(3
3) - nstin(32) - nstin(31) - nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(
25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin
(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nsti
n(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(40) = nstin(40) - nstin(39) - nstin(38) - nstin(37) - nstin(36) - nstin(35) - nstin(3
4) - nstin(33) - nstin(32) - nstin(31) - nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(
26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin
(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nsti
n(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) -
 nstin(1)

    'send "nstin"(input data) to the database input table here in the form of weight retained o
n each sieve size
    DataInput nstin, nsizes, 1, MyDb

    'to obtain several output data based on crusher settings for a given input
    For j = 1 To 10

        sset = SetArray(j)
        'call the crush routine
        result = crush(nstin(0), sset, nsout(0), ntype, itype, nsizes(0), nsieve)
        'the next three lines are for checking why the "CRUSH" routine does not work when debug
ging
        'result = 0 success
        ' -1,-2,-3 errors
        'MsgBox result

        'send "nsout" to the database output table here in the form of weight retained on each
sieve size
        DataOutput nsout, nsizes, sset, 1, MyDb
    Next
Next
MsgBox "Data processing is complete !"
Unload nnInput
End Sub
```

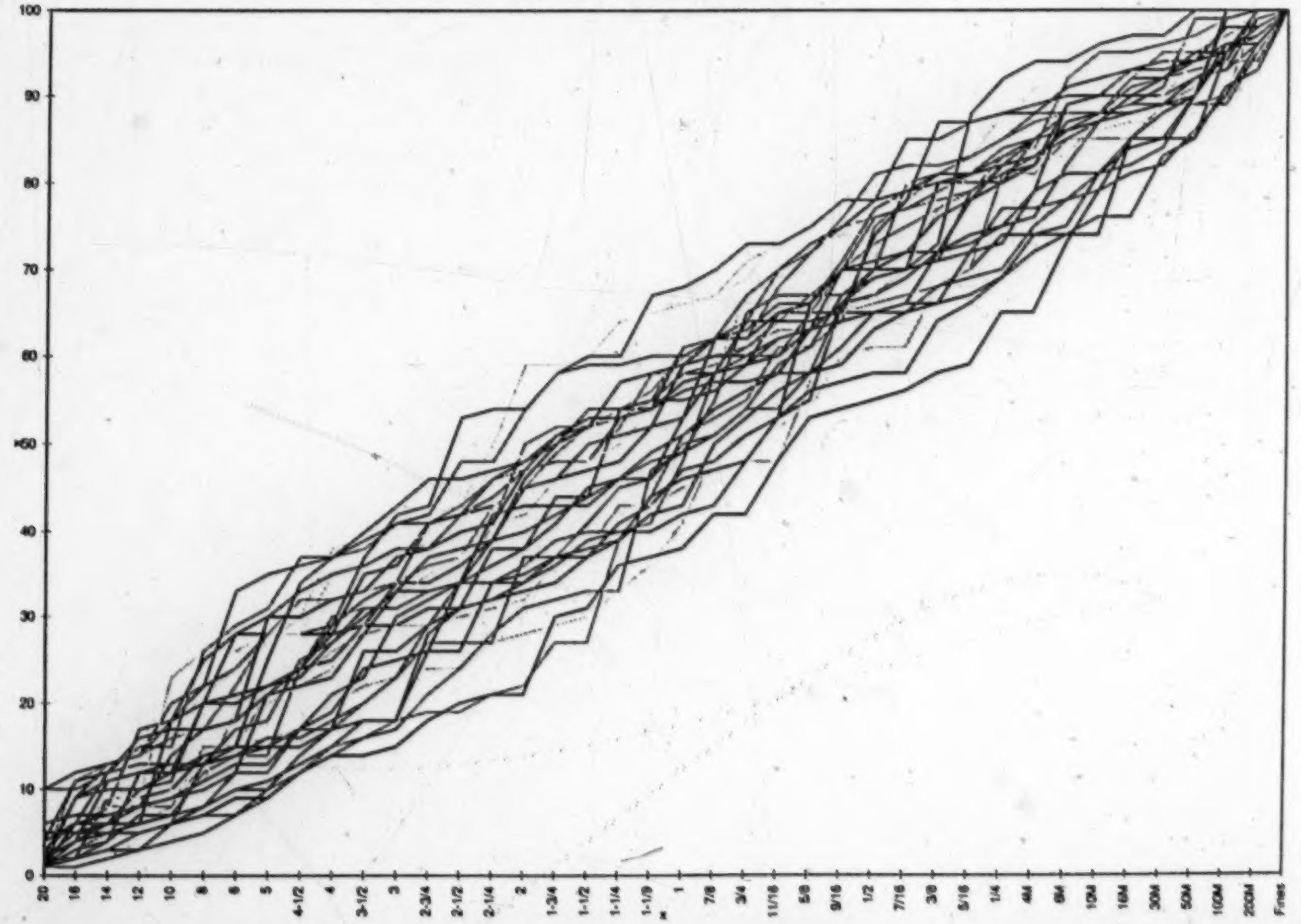
```
Option Explicit

'Description: stores the data processed by the "CRUSH" routine in the form of weight retained o
n each sieve size
'Input: nsout array of output, # of sieve sizes, # of samples to be generated, database locatio
n
'Output: Fills in the OUTPUT table in an ACCESS database with the output training data
'Side Effects: no global variables are changed by this function

Function DataOutput(nsout() As Single, nsizes() As Single, sset As Single, l As Integer, MyDb A
s Database)

'declare variables

Dim RSOutput As Recordset
Dim j As Integer

'open the temp table
Set RSOutput = MyDb.OpenRecordset("Output", dbOpenTable)

'now add the processed data to the output table
For j = 1 To 40
    RSOutput.AddNew
    RSOutput![Sample#] = l
    RSOutput![Setting] = sset
    RSOutput![Size] = nsizes(j)
    RSOutput![Retained] = nsout(j)
    RSOutput.Update
Next

'close the table
RSOutput.Close

End Function


'Description: stores the sorted data that has been converted into weight retained on each sieve
'Input: nstin arrayof input, # of sieve sizes, # of samples to be generated, database location
'Output: Fills in the INPUT table in an ACCESS database with the NN training data
'Side Effects: no global variables are changed by this function

Function DataInput(nstin() As Single, nsizes() As Single, l As Integer, MyDb As Database)

'declare variables
Dim RSInput As Recordset
Dim i As Integer
Dim j As Integer

'open the input table
Set RSInput = MyDb.OpenRecordset("Input", dbOpenTable)

'now add the input
For j = 1 To 40
    RSInput.AddNew
    RSInput![Sample#] = l
    RSInput![Size] = nsizes(j)
    RSInput![Retained] = nstin(j)
    RSInput.Update
Next

'close the table
RSInput.Close

End Function

'Description: stores the random numbers generated for each sample and sieve size
'Input: nstin arrayof input, # of sieve sizes, # of samples to be generated, database location
```

DBstorage - 2

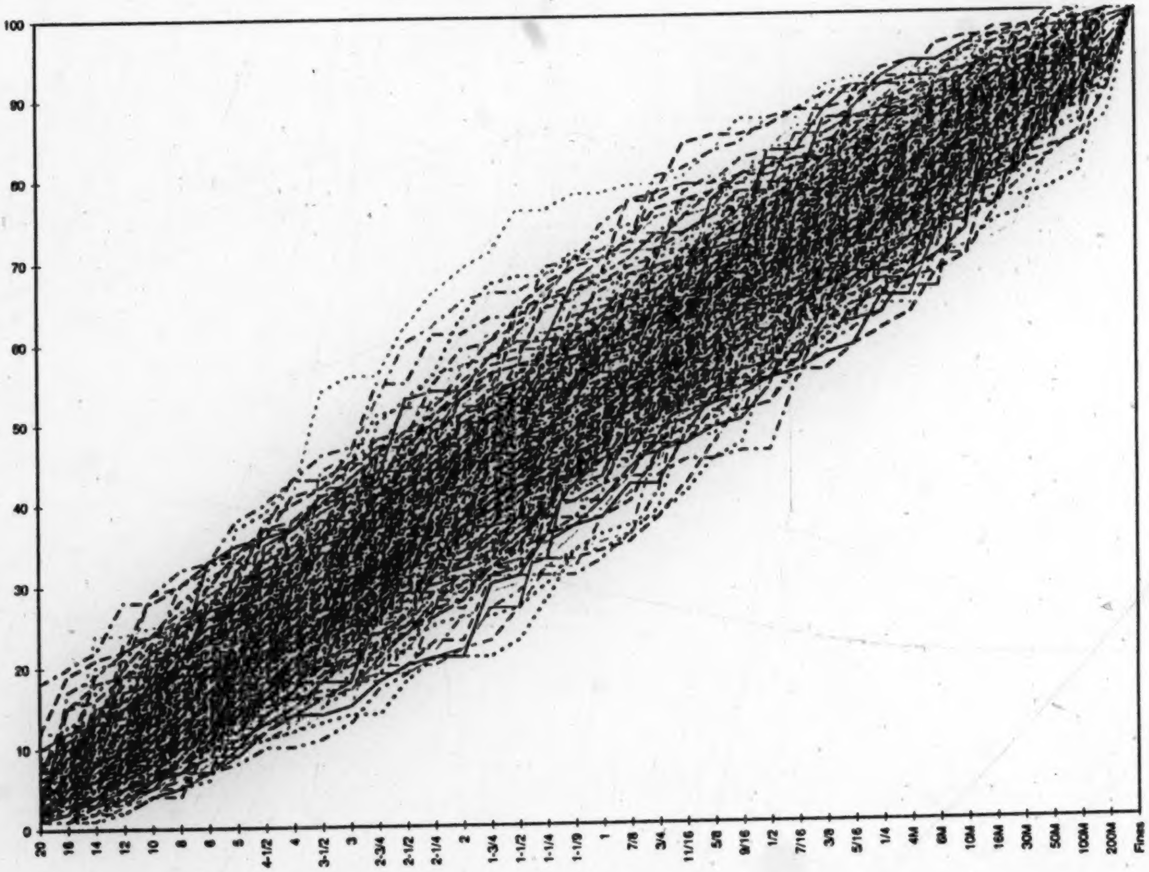
```
'Output: Fills in the TEMP table in an ACCESS database with the random #s
'Side Effects: no global variables are changed by this function

Function DataTemp(ByRef nstin() As Single, ByRef nsizes() As Single, l As Integer, MyDb As Data
base)

'declare variables
Dim RSTemporary As Recordset
Dim i As Integer
Dim j As Integer
Dim SQLtemp As String
Dim numcount As Integer

'open the temp table
Set RSTemporary = MyDb.OpenRecordset("Temporary", dbOpenTable)

'now add the random number input

For j = 1 To 40
    RSTemporary.AddNew
    RSTemporary![Sample#] = 1
    RSTemporary![Size] = nsizes(j)
    RSTemporary![Retained] = nstin(j)
    RSTemporary.Update
Next

RSTemporary.Close
'sort the data in ACCESS
SQLtemp = "SELECT DISTINCTROW Temporary.[Retained] "
SQLtemp = SQLtemp & "From Temporary Where Temporary.[sample#]= " & l
SQLtemp = SQLtemp & " ORDER BY Temporary.[Retained];"

Set RSTemporary = MyDb.OpenRecordset(SQLtemp, dbOpenDynaset)
RSTemporary.MoveFirst
RSTemporary.MoveLast
numcount = RSTemporary.RecordCount
RSTemporary.MoveFirst

'send the data back to VB
For j = 1 To numcount
    nstin(j) = RSTemporary![Retained]
    If j < numcount - 1 Then RSTemporary.MoveNext
Next

'close the table
RSTemporary.Close
End Function
```

APPENDIX P

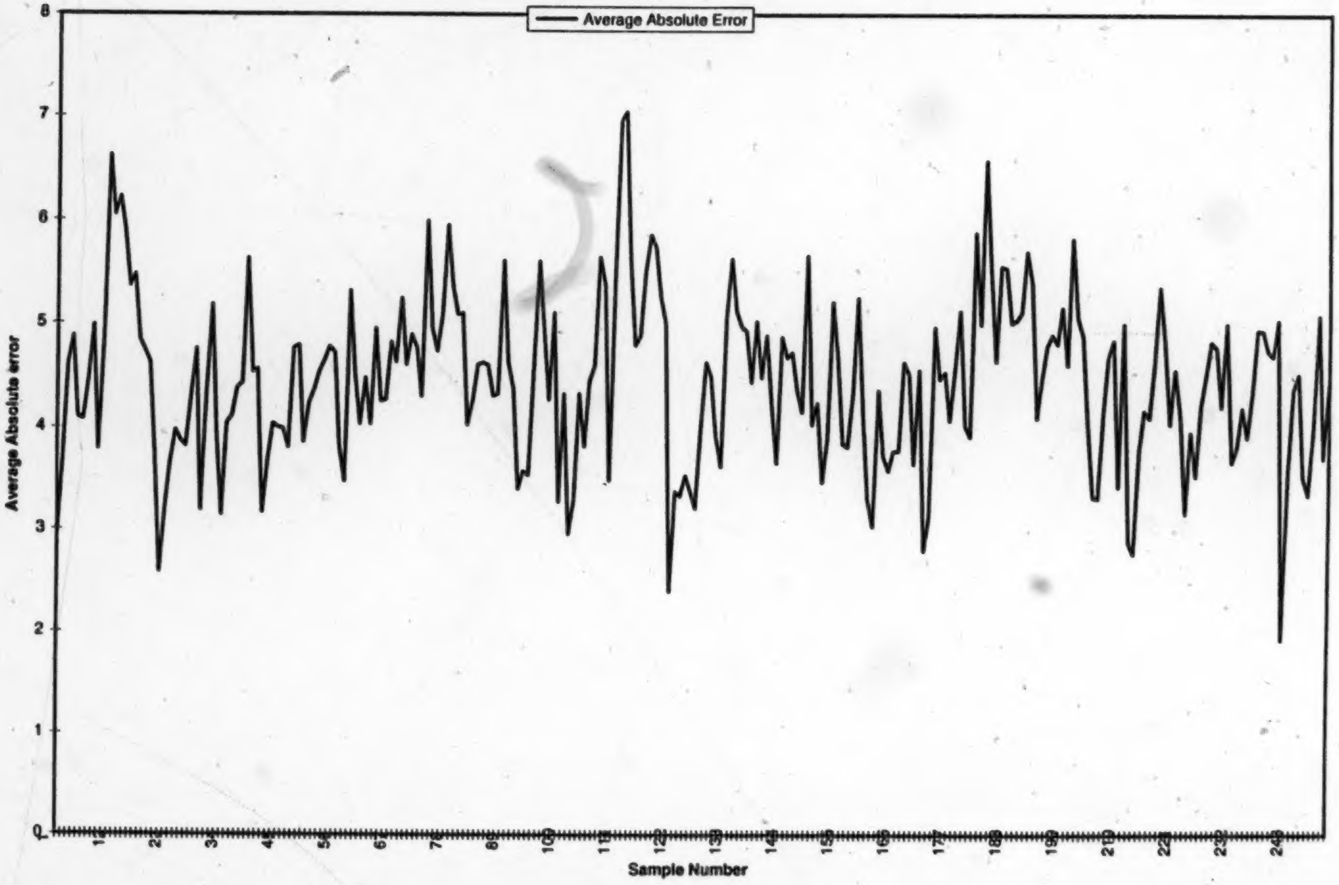
25 Sample Gradations Using Random Input0



250 Gradation Samples From Random Generated Data



Average Absolute Error (Weight Retained) For 250 Test Samples



APPENDIX Q

```
Option Explicit

Private Declare Function crush Lib "numrec.dll" (nstin As Single, ByVal sset As
Single, nsout As Single, ByVal ntype As Single, ByVal itype As Single, SIZES As
Single, ByVal nsieve As Single) As Integer




Private Sub cmdrun_Click()
'This version of the program is for a Cone Crusher
'Description: generates random numbers, utilizes the "CRUSH" routine, and create
s NN input and output for training and testing purposes
'Input: a number of different random combinations to be sent to the "CRUSH" rout
ine for all ten crusher settings
'Output: NN input and NN output for training and testing purposes
'Side Effects: the nstin and nsout arrays are passed to other functions and chan
ge for each set of different random numbers

' Declare Variables
    Dim nstin(0 To 41) As Single
    Dim sset As Single
    Dim nsout(0 To 41) As Single
    Dim ntype As Single
    Dim itype As Single
    Dim nsizes(0 To 41) As Single
    Dim nsieve As Single
    Dim result As Integer
    Dim i As Integer
    Dim j As Integer
    Dim k As Integer
    Dim l As Integer
    Dim Sum As Single
    Dim MyDB As Database
    Dim SetArray(0 To 10) As Single

    Dim LowerBound As Integer
    Dim UpperBound As Integer
    Dim NumSamples As Integer

    'Dim m As Integer
    'Dim n As Integer

' Initialize the Boundaries and Number of Samples Desired
    LowerBound = 1
    UpperBound = 41
    NumSamples = 0
    'where m is equal to the number of samples for each gradient
    'm = 20
' Initialize Parameters (leave ntype and itype as 1)
' Ntype:  1=Jaw, 2=Roll, 3=Cone, 4=Single Impeller, 5=Double Impeller
    'sset = Val(txtset.Text)
    ntype = 3
    itype = 1
    nsieve = 19

' Initialize the Sieve Size Array

    nsizes(1) = 20
    nsizes(2) = 16
    nsizes(3) = 14
    nsizes(4) = 12
    nsizes(5) = 10
    nsizes(6) = 8
    nsizes(7) = 6
    nsizes(8) = 5
    nsizes(9) = 4.5
    nsizes(10) = 4
    nsizes(11) = 3.5
    nsizes(12) = 3
    nsizes(13) = 2.75
    nsizes(14) = 2.5
    nsizes(15) = 2.25
    nsizes(16) = 2
    nsizes(17) = 1.75
    nsizes(18) = 1.5
    nsizes(19) = 1.25
    nsizes(20) = 1.11111
    nsizes(21) = 1
    nsizes(22) = 0.875
    nsizes(23) = 0.75
    nsizes(24) = 0.6875
    nsizes(25) = 0.625
    nsizes(26) = 0.5625
    nsizes(27) = 0.5
    nsizes(28) = 0.4375
    nsizes(29) = 0.375
    nsizes(30) = 0.3125
    nsizes(31) = 0.25
    'all the imperial "M sizes" come after this line
    nsizes(32) = 0.187
    nsizes(33) = 0.132
    nsizes(34) = 0.0787
    nsizes(35) = 0.0469
    nsizes(36) = 0.0234
    nsizes(37) = 0.0117
    nsizes(38) = 0.0059
    nsizes(39) = 0.0029
    'nsizes(40) = 0.000001

'set the number of data sets to be created
'k = Val(txtnumbset.Text)
'For l = 1 To k

'set the database for data storage
'Set MyDb = DBEngine.Workspaces(0).OpenDatabase("d:\thesis\vbaccess\generate\nnd
ata.mdb")
Set MyDB = DBEngine.Workspaces(0).OpenDatabase("d:\thesis\vbaccess\generate\nnda
ta3b.mdb")
'MsgBox "Passed"

'initializing the setting array

    SetArray(1) = 2
    SetArray(2) = 1.75
    SetArray(3) = 1.5
    SetArray(4) = 1.25
    SetArray(5) = 1
    SetArray(6) = 0.875
    SetArray(7) = 0.75
    SetArray(8) = 0.5
    SetArray(9) = 0.4375
    SetArray(10) = 0.375

'set the number of data sets to be created
'set k to 500 when actual data is to be obtained
'set k to 651* the number of samples desired over one spectrum of gradation orie
ntations(say 20,so 14000)
'k = 14000

'For n = 1 To m

'k = 14000
k = 100
```

```
nnInput - 3

'I am expecting 13020 samples to be generated

For l = 1 To k

    'obtaining random numbers for %retained for each sieve
        For i = 1 To 40
            nstin(i) = 1000 * Rnd()
        Next

    'data is sent to a temporary table to sort the data
    DataTemp nstin, nsizes, l, MyDB

'translate the cumulative weight retained to actual weight retained on each siev
e

    nstin(1) = nstin(1)
    nstin(2) = nstin(2) - nstin(1)
    nstin(3) = nstin(3) - nstin(2) - nstin(1)
    nstin(4) = nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(5) = nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(6) = nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(7) = nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) -
 nstin(1)
    nstin(8) = nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) -
 nstin(2) - nstin(1)
    nstin(9) = nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) -
 nstin(3) - nstin(2) - nstin(1)
    nstin(10) = nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5)
 - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(11) = nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6
) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(12) = nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(
7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(13) = nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin
(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(14) = nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nsti
n(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2
) - nstin(1)
    nstin(15) = nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nsti
n(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(
3) - nstin(2) - nstin(1)
    nstin(16) = nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nsti
n(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin
(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(17) = nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nsti
n(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nsti
n(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(18) = nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nsti
n(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nst
in(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(19) = nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nsti
n(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - ns
tin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(20) = nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nsti
n(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - n
stin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nsti
n(1)
    nstin(21) = nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nsti
n(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) -
nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nst
in(2) - nstin(1)
    nstin(22) = nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nsti
n(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) -
nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - ns
tin(3) - nstin(2) - nstin(1)
    nstin(23) = nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nsti
n(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) -
nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - n
```

```
stin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(24) = nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nsti
n(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) -
nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) -
nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(25) = nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nsti
n(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) -
nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) -
 nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(26) = nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nsti
n(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) -
nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8)
- nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(27) = nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nsti
n(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) -
nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9)
 - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) -
nstin(1)
    nstin(28) = nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nsti
n(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) -
nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10
) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) -
 nstin(2) - nstin(1)
    nstin(29) = nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(25) - nsti
n(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) -
nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11
) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4)
- nstin(3) - nstin(2) - nstin(1)
    nstin(30) = nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(26) - nsti
n(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) -
nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12
) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5)
 - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(31) = nstin(31) - nstin(30) - nstin(29) - nstin(28) - nstin(27) - nsti
n(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) -
nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13
) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6
) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(32) = nstin(32) - nstin(31) - nstin(30) - nstin(29) - nstin(28) - nsti
n(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) -
nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14
) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(
7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(33) = nstin(33) - nstin(32) - nstin(31) - nstin(30) - nstin(29) - nsti
n(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) -
nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15
) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin
(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(34) = nstin(34) - nstin(33) - nstin(32) - nstin(31) - nstin(30) - nsti
n(29) - nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) -
nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16
) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nsti
n(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2
) - nstin(1)
    nstin(35) = nstin(35) - nstin(34) - nstin(33) - nstin(32) - nstin(31) - nsti
n(30) - nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) -
nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17
) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nsti
n(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(
3) - nstin(2) - nstin(1)
    nstin(36) = nstin(36) - nstin(35) - nstin(34) - nstin(33) - nstin(32) - nsti
n(31) - nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(26) - nstin(25) -
nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19) - nstin(18
) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nstin(12) - nsti
n(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nstin(5) - nstin
(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(37) = nstin(37) - nstin(36) - nstin(35) - nstin(34) - nstin(33) - nsti
n(32) - nstin(31) - nstin(30) - nstin(29) - nstin(28) - nstin(27) - nstin(26) -
```

```
nnInput - 5

nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20) - nstin(19
) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nstin(13) - nsti
n(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nstin(6) - nsti
n(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(38) = nstin(38) - nstin(37) - nstin(36) - nstin(35) - nstin(34) - nsti
n(33) - nstin(32) - nstin(31) - nstin(30) - nstin(29) - nstin(28) - nstin(27) -
nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21) - nstin(20
) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nstin(14) - nsti
n(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - nstin(7) - nst
in(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(39) = nstin(39) - nstin(38) - nstin(37) - nstin(36) - nstin(35) - nsti
n(34) - nstin(33) - nstin(32) - nstin(31) - nstin(30) - nstin(29) - nstin(28) -
nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22) - nstin(21
) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nstin(15) - nsti
n(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - nstin(8) - ns
tin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nstin(1)
    nstin(40) = nstin(40) - nstin(39) - nstin(38) - nstin(37) - nstin(36) - nsti
n(35) - nstin(34) - nstin(33) - nstin(32) - nstin(31) - nstin(30) - nstin(29) -
nstin(28) - nstin(27) - nstin(26) - nstin(25) - nstin(24) - nstin(23) - nstin(22
) - nstin(21) - nstin(20) - nstin(19) - nstin(18) - nstin(17) - nstin(16) - nsti
n(15) - nstin(14) - nstin(13) - nstin(12) - nstin(11) - nstin(10) - nstin(9) - n
stin(8) - nstin(7) - nstin(6) - nstin(5) - nstin(4) - nstin(3) - nstin(2) - nsti
n(1)

    ' decide on what to zero
    If NumSamples < 19 Then
            NumSamples = NumSamples + 1

        Else
        ' Increment the lowerbound value of which the upperbound value depends u
pon
        LowerBound = LowerBound + 1

            If LowerBound = UpperBound - 4 Then
                LowerBound = 1
                UpperBound = UpperBound - 1
            End If

        NumSamples = 0
    End If
        ' for lowerbound control of sieve sizes to set to zero
        For i = 1 To LowerBound
            nstin(i) = 0
        Next

        ' for upperbound control of sieve sizes to set to zero
        For i = 40 To UpperBound Step -1
            nstin(i) = 0
        Next

        ' Increment the lowerbound value of which the upperbound value depends u
pon
        'LowerBound = LowerBound + 1

          '   If LowerBound = UpperBound - 4 Then
           '      LowerBound = 1
            '     UpperBound = UpperBound - 1
             'End If


     'send "nstin" to the database input table here
     DataInput nstin, nsizes, 1, MyDB

     'to obtain several output data based on crusher settings for a given input
     For j = 1 To 10

         sset = SetArray(j)
```

```
nnInput - 6
        'call the crush routine
               result = crush(nstin(0), sset, nsout(0), ntype, itype, nsizes(0),
 nsieve)
        'the next three lines are for checking why the crush routine does not wo
rk when debugging
        'result = 0 success
        ' -1,-2,-3 errors
        'MsgBox result

        'send "nsout" to the database output table here in the form of weight re
tained on each sieve size
        DataOutput nsout, nsizes, sset, 1, MyDB
    Next

    ' to terminate the orientation of sampling and the entire process if k = 651
* number of samples desired for each gradation orientation line
    If UpperBound = 10 Then Exit For

Next
'Next
MsgBox "Data processing is complete !"
Unload nnInput
End Sub
```

DBstorage - 1

```
Option Explicit
Function DataOutput(nsout() As Single, nsizes() As Single, sset As Single, l As
Integer, MyDB As Database)
'Description: stores the data processed by the "CRUSH" routine in the form of we
ight retained on each sieve size
'Input: nsout array of output, # of sieve sizes, # of samples to be generated, a
nd the database location
'Output: fills in the OUTPUT table in an ACCESS database with the output trainin
g data
'Side Effects: no global variables are changed by this function

'declare variables
'Dim MyDb As Database
Dim RSOutput As Recordset
Dim j As Integer

'add the output into the output table
'SetDatabase MyDb
'Set MyDb = DBEngine.Workspaces(0).OpenDatabase("c:\civil96\civ603\project\nndat
a.mdb")

'open the temp (temporary table and enter the data
Set RSOutput = MyDB.OpenRecordset("Output", dbOpenTable)

'now add the processed data to the output table
For j = 1 To 40
    RSOutput.AddNew
    RSOutput![Sample#] = l
    RSOutput![Setting] = sset
    RSOutput![Size] = nsizes(j)
    RSOutput![Retained] = nsout(j)
    RSOutput.Update
Next

'close the table
RSOutput.Close

End Function


Function DataInput(nstin() As Single, nsizes() As Single, l As Integer, MyDB As
Database)
'Description: stores the sorted data that has been converted into weight retaine
d on each sieve size
'Input: nsin array of input, # of sieve sizes, # of samples to be generated,and
database location
'Output: Fills in the INPUT table in an ACCESS database with the NN input traini
ng data
'Side Effects: no global variables are changed by this function

'declare variables
'Dim MyDb As Database
Dim RSInput As Recordset
Dim i As Integer
Dim j As Integer

 'add the sorted input into the input table
 'SetDatabase MyDb
 'Set MyDb = DBEngine.Workspaces(0).OpenDatabase("c:\civil96\civ603\project\nndat
 a.mdb")

 'open the input table and enter the data
 Set RSInput = MyDB.OpenRecordset("Input", dbOpenTable)

 'now add the input
 For j = 1 To 40
```

```
    RSInput.AddNew
    RSInput![Sample#] = 1
    RSInput![Size] = nsizes(j)
    RSInput![Retained] = nstin(j)
    RSInput.Update
Next

'close the table
RSInput.Close

End Function
Function DataTemp(ByRef nstin() As Single, ByRef nsizes() As Single, l As Intege
r, MyDB As Database)
'Description: stores the random numbers generated for each sample and sieve size
'Input: nstin array of input, # of sieve size, # of samples to be generated, and
 the database location
'Output: fills in the TEMP table in an ACCESS database with the random numbers
'Side Effects: no global variables are changed by this function

'declare variables
'Dim MyDb As Database
Dim RSTemporary As Recordset
Dim i As Integer
Dim j As Integer
Dim SQLtemp As String
Dim numcount As Integer
'add the random input into the temp table
'SetDatabase MyDb
'Set MyDb = DBEngine.Workspaces(0).OpenDatabase("c:\civil96\civ603\project\nndat
a.mdb")

'open the temp table and enter the data
Set RSTemporary = MyDB.OpenRecordset("Temporary", dbOpenTable)

'now add the random number input

For j = 1 To 40
    RSTemporary.AddNew
    RSTemporary![Sample#] = 1
    RSTemporary![Size] = nsizes(j)
    RSTemporary![Retained] = nstin(j)
    RSTemporary.Update
Next

'close the table
RSTemporary.Close

'sort the data in ACCESS
SQLtemp = "SELECT DISTINCTROW Temporary.[Retained] "
SQLtemp = SQLtemp & "From Temporary Where Temporary.[sample#]= " & l
SQLtemp = SQLtemp & " ORDER BY Temporary.[Retained];"

Set RSTemporary = MyDB.OpenRecordset(SQLtemp, dbOpenDynaset)
RSTemporary.MoveFirst
RSTemporary.MoveLast
numcount = RSTemporary.RecordCount
RSTemporary.MoveFirst

'send the data back to Visual Basic
For j = 1 To numcount
        nstin(j) = RSTemporary!Retained
    If j < numcount - 1 Then RSTemporary.MoveNext
Next

'close the table
RSTemporary.Close
End Function
```

APPENDIX R

Neural Network Weights For Prototype Crushing Neural Network

Node Number	Layer	Weight
1 = Bias Node	N/A	0
2 = Crusher Setting	Input	0
3	"	0
4	"	0
5	"	0
6	"	0
7	"	0
8	"	0
9	"	0
10	"	0
11	"	0
12	"	0
13	"	0
14	"	0
15	"	0
16	"	0
17	"	0
18	"	0
19	"	0
20	"	0
21	"	0
22	"	0
23	"	0
24	"	0
25	"	0
26	"	0
27	"	0
28	"	0
29	"	0
30	"	0
31	"	0
32	"	0
33	"	0
34	"	0
35	"	0
36	"	0
37	"	0
38	"	0
39	"	0
40	"	0

41	“	0
42	“	0
43	Hidden	-0.139563
44	“	-0.517306
45	“	-0.158918
46	“	-0.421807
47	“	-0.150705
48	“	-0.852999
49	“	-0.439972
50	“	-0.196728
51	“	-0.153452
52	“	-0.425756
53	“	-0.158896
54	“	-0.109626
55	“	-1.068437
56	“	-0.129131
57	“	-0.125322
58	“	0.005145
59	“	-0.166987
60	“	-0.185345
61	“	-0.041228
62	“	-0.160466
63	Output	0.000707
64	“	0.001051
65	“	0.001065
66	“	-0.000714
67	“	0.000929
68	“	-0.001313
69	“	-0.009024
70	“	-0.002218
71	“	0.000425
72	“	0.000337
73	“	0.003365
74	“	-1.660398
75	“	-1.659711
76	“	-1.489924
77	“	-0.314595
78	“	-0.803267
79	“	0.542023
80	“	1.069307
81	“	0.928765
82	“	0.322698
83	“	1.098145
84	“	-0.34626

85	“	-0.313396
86	“	0.680092
87	“	-0.708380
88	“	-1.126101
89	“	-0.576094
90	“	-1.374429
91	“	-0.662417
92	“	-0.465843
93	“	-0.780161
94	“	-1.159663
95	“	-1.049586
96	“	-0.748155
97	“	-1.222885
98	“	-0.478078
99	“	-1.297313
100	“	-0.901520
101	“	-1.476395
102	“	-0.6594315

Neural Network Weights For Final Crushing Neural Network

From Bias To Middle Layer Nodes

	Bias Node
Middle Node #1	[illegible]
Middle Node #2	[illegible]
Middle Node #3	[illegible]
Middle Node #4	[illegible]
Middle Node #5	[illegible]
Middle Node #6	[illegible]
Middle Node #7	[illegible]
Middle Node #8	[illegible]
Middle Node #9	[illegible]
Middle Node #10	[illegible]
Middle Node #11	[illegible]
Middle Node #12	[illegible]
Middle Node #13	[illegible]
Middle Node #14	[illegible]
Middle Node #15	[illegible]
Middle Node #16	[illegible]
Middle Node #17	[illegible]
Middle Node #18	[illegible]
Middle Node #19	[illegible]
Middle Node #20	[illegible]
Middle Node #21	[illegible]
Middle Node #22	[illegible]
Middle Node #23	[illegible]
Middle Node #24	[illegible]
Middle Node #25	[illegible]
Middle Node #26	[illegible]
Middle Node #27	[illegible]
Middle Node #28	[illegible]
Middle Node #29	[illegible]
Middle Node #30	[illegible]

From Input To Middle Layer Nodes

	Input #1	Input #2	Input #3	Input #4	Input #5	Input #6	Input #7	Input #8	Input #9	Input #10	Input #11	Input #12	Input #13	Input #14	Input #15	Input #16	Input #17	Input #18	Input #19	Input #20	Input #21
Middle Node #1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #2	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #3	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #4	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #5	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #6	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #7	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #8	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #9	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #10	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #11	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #12	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #13	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #14	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #15	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #16	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #17	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #18	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #19	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #20	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #21	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #22	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #23	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #24	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #25	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #26	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #27	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #28	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #29	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Middle Node #30	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

From Bias To Output Layer Nodes

	Bias Node
Output Node #1	[illegible]
Output Node #2	[illegible]
Output Node #3	[illegible]
Output Node #4	[illegible]
Output Node #5	[illegible]
Output Node #6	[illegible]
Output Node #7	[illegible]
Output Node #8	[illegible]

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
79	Output Node #9	[illegible]																				
80	Output Node #10	[illegible]																				
81	Output Node #11	[illegible]																				
82	Output Node #12	[illegible]																				
83	Output Node #13	[illegible]																				
84	Output Node #14	[illegible]																				
85	Output Node #15	[illegible]																				
86	Output Node #16	[illegible]																				
87	Output Node #17	[illegible]																				
88	Output Node #18	[illegible]																				
89	Output Node #19	[illegible]																				
90	Output Node #20	[illegible]																				
91	Output Node #21	[illegible]																				
92	Output Node #22	[illegible]																				
93	Output Node #23	[illegible]																				
94	Output Node #24	[illegible]																				
95	Output Node #25	[illegible]																				
96	Output Node #26	[illegible]																				
97	Output Node #27	[illegible]																				
98	Output Node #28	[illegible]																				
99	Output Node #29	[illegible]																				
100	Output Node #30	[illegible]																				
101	Output Node #31	[illegible]																				
102	Output Node #32	[illegible]																				
103	Output Node #33	[illegible]																				
104	Output Node #34	[illegible]																				
105	Output Node #35	[illegible]																				
106	Output Node #36	[illegible]																				
107	Output Node #37	[illegible]																				
108	Output Node #38	[illegible]																				
109	Output Node #39	[illegible]																				
110	Output Node #40	[illegible]																				
111																						
112	**From Input To Middle Layer Nodes**																					
113		Middle #1	Middle #2	Middle #3	Middle #4	Middle #5	Middle #6	Middle #7	Middle #8	Middle #9	Middle #10	Middle #11	Middle #12	Middle #13	Middle #14	Middle #15	Middle #16	Middle #17	Middle #18	Middle #19	Middle #20	Middle #21
114	Output Node #1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
115	Output Node #2	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
116	Output Node #3	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
117	Output Node #4	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
118	Output Node #5	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
119	Output Node #6	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
120	Output Node #7	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
121	Output Node #8	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
122	Output Node #9	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
123	Output Node #10	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
124	Output Node #11	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
125	Output Node #12	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
126	Output Node #13	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
127	Output Node #14	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
128	Output Node #15	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
129	Output Node #16	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
130	Output Node #17	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
131	Output Node #18	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
132	Output Node #19	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
133	Output Node #20	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
134	Output Node #21	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
135	Output Node #22	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
136	Output Node #23	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
137	Output Node #24	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
138	Output Node #25	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
139	Output Node #26	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
140	Output Node #27	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
141	Output Node #28	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
142	Output Node #29	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
143	Output Node #30	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
144	Output Node #31	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
145	Output Node #32	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
146	Output Node #33	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
147	Output Node #34	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
148	Output Node #35	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
149	Output Node #36	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
150	Output Node #37	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
151	Output Node #38	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
152	Output Node #39	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
153	Output Node #40	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO
37	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]	Input [illegible]
38	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
39	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
40	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
41	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
42	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
43	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
44	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
45	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
46	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
47	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
48	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
49	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
50	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
51	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
52	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
53	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
54	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
55	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
56	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
57	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
58	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
59	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
60	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
61	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
62	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
63	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
64	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
65	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
66	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
67	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

	W	X	Y	Z	AA	AB	AC	AD	AE
113	Middle #72	Middle #73	Middle #74	Middle #75	Middle #76	Middle #77	Middle #78	Middle #79	Middle #80
114	-4.0001282	-1.2073046	-0.3345533	1.1884284	-0.32347	0.14707030	-0.1507952	-0.888874	-0.9909884
115	-3.37779705	-0.0013198	-0.530703	0.78504907	-0.7090274	-1.1793378	0.21550249	-0.4908302	-0.60178
116	0.35502195	-0.949153	-0.924081	0.8730016	0.1070122	-0.5111916	1.0311442	-1.4372981	0.05804082
117	0.4829813	-2.2400128	-0.077904	-0.00411511	0.0719015	-0.8265384	-0.4572458	0.30643544	-1.2712255
118	0.54880544	-0.1900502	-1.1190061	2.4582816	-1.1958877	2.0020900	1.7859784	1.4181134	-0.1970077
119	0.09742273	0.7955149	1.0494305	-1.4143493	-0.0033481	-0.3367248	0.74173045	1.3037254	-1.440198
120	-2.405674	-3.7142045	-0.5141171	1.4240578	0.10085027	3.0211118	-0.1040046	-1.0091974	-1.0036455
121	-1.4584495	0.00011282	-0.6570905	-0.6292234	0.47090829	-0.4782458	0.04058751	-1.4200089	1.8303702
122	-2.0581728	-2.1300097	-1.2507728	0.9907372	-0.3784387	2.5204109	-0.0070518	-1.9343743	0.0099003
123	1.9057583	-0.0409008	0.50917085	-1.5117816	0.1002942	-0.2795158	0.3179335	-0.5522635	-0.3023044
124	0.340511184	3.4995047	0.67253084	1.9174835	-0.825738	-2.1810900	0.82517240	0.21078255	-0.8116427
125	-0.51205581	0.1082848	0.70302082	-0.07411775	0.70874318	0.70375134	1.2532420	1.6101080	-0.7715834
126	0.713327717	1.6330846	0.71849751	-0.9830054	-0.4359705	-1.0900703	1.0113128	1.2262777	0.01100071
127	1.6204050	0.70905084	0.24575401	0.07981348	-0.1095079	-0.0811750	2.531510	-1.6238474	0.41175562
128	0.21484908	-2.9122241	0.24018185	-3.2863138	-0.4829788	1.0870824	-0.5395022	1.5750557	0.43090075
129	1.3652540	2.8277714	0.95570040	1.6180550	0.70005028	-1.6184080	0.56281675	0.47205802	1.4250001
130	-0.050091	-0.0084482	-0.1499108	2.6732457	-0.61518051	0.091117492	0.2807742	0.41142957	-0.2485283
131	0.07774092	1.0403079	1.0000036	0.50353509	-0.0981471	-1.8182338	-1.4410800	-0.3832400	0.0800000
132	-2.4053003	0.62948545	0.10503513	1.1825334	0.24857517	2.6301165	0.09782457	-3.5504477	-0.1340041
133	1.5004636	2.0750835	2.7711116	-2.2304197	1.741082	2.0047130	-0.6039230	-1.7804783	1.1009305
134	-1.1474946	2.0013656	0.24175324	-1.2790570	-1.5510712	-1.7005007	-0.7305040	-1.0901425	-1.3117621
135	-2.0215631	0.41200016	-1.0111714	-0.3040047	0.49002348	0.04300008	0.40034871	0.75204618	0.05300779
136	-2.77483399	0.45400372	1.3140700	1.9251016	-1.0205034	2.0123117	-1.4002507	-2.4788195	2.1027258
137	2.0532051	-2.7954019	1.5370612	-0.3014383	-1.7133587	1.6300007	-0.1014477	-1.7010100	1.3445003
138	2.3401119	-2.4378218	1.0582722	-2.300215	-1.7830737	-0.2711008	-0.4108356	0.70990050	-0.551040
139	0.30053000	1.00030	0.04370571	-1.0431235	-0.1835500	-1.4008133	0.17300109	1.1067243	-0.0033340
140	1.1491071	-1.3023381	1.1405272	0.90023287	-0.2033782	0.57700503	1.1670049	-0.5004341	-1.3032308
141	0.67054336	1.7167512	-0.2000730	0.5817247	-0.3472458	-0.5370000	-0.9121046	0.62140072	2.316119
142	-1.0041527	-0.3082436	2.0205107	0.40250046	-1.0800750	0.70332482	0.33305043	0.04797415	1.1921413
143	-1.7004047	-1.2752000	-1.0514159	1.0111222	0.70041177	2.3154912	-1.1014040	0.00090076	0.11740400
144	-0.0400013	0.70023219	-0.3180104	4.3230497	0.09724084	0.15027677	3.0357027	-2.3743500	0.49795047
145	-4.6530704	-0.2500091	-1.7024703	-1.7002075	-0.3714303	0.05237031	-1.9010002	-0.6504205	1.350917
146	3.3079514	1.0325519	0.97036756	1.424708	0.9005747	1.0501301	1.1022497	0.70518461	0.5791700
147	3.0267747	-4.317796	-0.0791083	-2.4456721	1.2190020	2.3072017	-0.7004914	1.7032470	0.40200740
148	3.9440900	-1.1000014	-0.100332	-0.1515032	0.74158511	2.004300	-1.1002003	2.4100490	-0.0050507
149	-0.95400032	-0.0472063	1.0030700	-0.182300	1.3002020	-0.3100540	-0.0200557	-2.4207023	0.37930811
150	2.81271857	0.50067054	0.74224972	-1.5210641	1.5174887	1.7052840	0.05090085	0.10527094	0.37371357
151	-1.5077003	0.65344097	-0.1121407	2.0082054	0.76787647	-1.5584502	1.1302500	-0.9987401	0.05329744
152	-1.1176700	1.2726804	-0.304055	-0.6250005	-2.0329241	1.3648833	0.69444727	-0.8537072	-2.6318165
153	[illegible]	-1.0020000	0.21724831	-2.0504504	[illegible]	-0.2248715	0.90125250	-0.7466552	1.1896251